P. 1877/92

6 1992

informatyka

WYDAWNICTWO SIGMA NOT

Cena 18 000 zł

informatyka

nr 5 1992
Maj
Rok wydania XXVII

Miesięcznik
ISSN 0542-9951
INDEKS 36124

KOLEGIUM REDAKCYJNE:

mgr Jarosław DEMINET
mgr inż. Piotr FUGLEWICZ
mgr Teresa JABŁOŃSKA
(sekretarz redakcji)
Władysław KLEPACZ
(redaktor naczelny)
dr inż. Wojciech MOKRZYCKI
mgr inż. Jan RYŻKO
dr Zdzisław SZYJEWSKI
mgr Hanna WŁODARSKA –
– MARCZENKO

PRZEWODNICZĄCY
RADY PROGRAMOWEJ:

Prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictwo Czasopism i Książek
Technicznych SIGMA NOT
Spółka z o.o.
ul. Biała 4
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
01-552 Warszawa,
Pl. Inwalidów 10, p. 104, 105
tel. 39-14-34

Materiałów nie zamówionych
redakcja nie zwraca

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji lub Działu Reklamy i Marketingu 00-950 Warszawa ul. Biała 4 telefon: 20-31-24 telefaks: 20-31-16 teleks: 814550

W numerze:

Strona

Laboratoryjny system mikrokomputerowy – *Jerzy R. Chrząszcz, Grzegorz B. Mazur* 1
Inteligentny system gromadzenia informacji faktograficznej – *Zdzisław Wrzeszcz* 9
Rekonfigurowalny filtr obrazów cyfrowych – *Dariusz Caban, Jacek Jarnicki* 15
APW – graficzne narzędzia programowania – *Jan Piecha, Andrzej Przybylski, Jarosław Zyguła* 20

/unix
Jakim jesteś unixowcem? – *Agnieszka Pietrusińska* 27

Ze świata
Informatyka w Singapurze – *Jan Ryżko* 28

W najbliższych numerach:

- **Paweł Podrecki pisze o ochronie prawnej topografii układów scalonych, której głównym zadaniem powinno być zabezpieczenie przed niekontrolowanym kopiowaniem produktów półprzewodnikowych.**
- **Małgorzata Byrska podaje wytyczne EWG w sprawie ochrony prawnej programów komputerowych i na ich tle omawia polski projekt prawa autorskiego.**
- **Zdzisław Szyjewski definiuje systemy otwarte oraz podaje przykłady rozwiązań problemu wg koncepcji firm Hewlett Packard, DEC i IBM.**
- **Katarzyna Lange charakteryzuje praktyczne wykorzystanie protokołu NetBIOS.**
- **Krzysztof Zieliński omawia stan i perspektywy rozwoju uczelnianej sieci komputerowej Akademii Górniczo-Hutniczej.**

Warunki prenumeraty

Przyjęcie prenumeraty – wyłącznie na podstawie dokonanej wpłaty na drukach dostarczanych dotychczasowym prenumeratorom przez Wydawnictwo, lub nowym – po uprzednim zgłoszeniu zapotrzebowania (pisemnie lub telefonicznie) w Zakładzie Kolportażu Wydawnictwa.

Blankiet wpłaty – powinien zawierać następujące informacje: dokładna nazwa i adres (z kodem pocztowym) zamawiającego, tytuły zamawianych czasopism, ich liczbę i okres prenumeraty.

Wpłata – zgodnie z podanymi cenami należy dokonać w banku lub w UPT na konto podane na naszym blankiecie, tj:
Państwowy Bank Kredytowy III O/Warszawa nr: 370015-1573-139-11

Prenumeratorzy zbiorowi – osoby prawne obowiązują blankiety „Wpłata-Zamówienie". Cena normalna.

Prenumeratorzy indywidualni – osoby fizyczne obowiązują blankiety typu przekazy dla wpłat na rachunki bankowe. Cena normalna.

Prenumerata ulgowa – zgodnie z podaną ceną ulgową przysługuje wyłącznie osobom fizycznym, będącym członkami SNT, studentom i uczniom szkół zawodowych. Uczniowie szkół ogólnokształcących mogą zamówić w prenumeracie ulgowej tylko miesięcznik „Aura"

Uwaga! w podanym okresie prenumeraty można zamówić tylko po jednym egzemplarzu z każdego tytułu.

Prenumerata ze zleceniem wysyłki za granicę – cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa od ceny normalnej.

Należy podać dokładny adres odbiorcy za granicą.

Terminy przyjmowania prenumeraty:
– do 10 listopada na I, II, III, IV kwartał następnego roku
– do 28 lutego na II, III, IV kwartał br.
– do 31 maja na III i IV kwartał br.
– do 31 sierpnia na IV kwartał br.

Zmiany w prenumeracie, np. zmiana liczby tytułów, liczby egzemplarzy, rezygnacja z prenumeraty, można zgłaszać tylko w podanych terminach z mocą obowiązującą od następnego kwartału.

Egzemplarze archiwalne (z lat ubiegłych)
Można nabyć za gotówkę w Klubie Prasy Technicznej, Warszawa, ul. Mazowiecka 12 (tel. 26-80-16) lub zamówić pisemnie w Zakładzie Kolportażu, Dział Handlowy, 00-950 Warszawa, skr. poczt. 1004 (tel. 40-37-31), na rachunek lub za zaliczeniem pocztowym.

Informacji o prenumeracie udziela: Zakład Kolportażu Wydawnictwa SIGMA-NOT Spółka z o.o., 00-716 Warszawa, ul. Bartycka 20, skr. 1004. Telefony: 40-00-21 wewn. 293, 295, 299 lub 40-30-86, 40-35-89.

Wstępna cena jednego egzemplarza na 1992 rok: normalna – 18 000 zł, ulgowa – 13 500zł

Wartość prenumeraty (w zł):
Normalna: kwartalna – 54 000, półroczna 108 000, roczna 216 000
Ulgowa: kwartalna – 40 500, półroczna 81 000, roczna 162 000

Uwaga: W przypadku zmiany cen w okresie objętym prenumeratą, prenumeratorzy zobowiązani są do dopłaty różnicy cen.

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 141/92

P. 1877/92

JERZY R. CHRZĄSZCZ
GRZEGORZ B. MAZUR
Instytut Informatyki
Politechnika Warszawska

Laboratoryjny system mikrokomputerowy

Laboratoryjny system mikroprocesorowy jest istotnym narzędziem pracy dydaktycznej, doświadczalnej i konstrukcyjnej w dziedzinie budowy systemów mikroprocesorowych. System taki powinien umożliwiać konstruowanie prototypów zróżnicowanych konfiguracji systemów mikroprocesorowych zarówno w celach demonstracyjnych i dydaktycznych, jak i dla modelowania konstrukcji nowych systemów do różnych zastosowań.

Z doświadczeń autorów wynika, że wymagania stawiane systemom dydaktycznym i uruchomieniowym są podobne. Dotyczą one przede wszystkim możliwości ingerowania w strukturę połączeń, dowolności w wyborze elementów oraz mechanizmów sterowania i monitorowania pracy systemu na różnych poziomach. W zastosowaniach dydaktycznych istotne jest także zbliżenie przebiegu zajęć laboratoryjnych do rzeczywistego procesu projektowania i uruchamiania systemu mikroprocesorowego. Umożliwia to studentowi nabycie w czasie ćwiczeń laboratoryjnych umiejętności niezbędnych w przyszłej pracy inżynierskiej. Jest więc wskazane, aby student mógł samodzielnie rozwiązywać problemy doboru zestawu elementów, określenia logicznej struktury szyny systemu, dołączenia poszczególnych elementów do systemu, dekodowania modułów w przestrzeni adresowej, synchronizacji itp.

Prowadząc od kilku lat zajęcia z zakresu architektury komputerów oraz projektowania i uruchamiania systemów mikroprocesorowych, autorzy mieli okazję do zebrania wielu doświadczeń i wniosków, które stały się podstawą do wprowadzanych obecnie zmian w programie nauczania tych przedmiotów i posłużyły do sformułowania koncepcji opisywanego systemu laboratoryjnego. W dotychczasowym przebiegu nauczania główny nacisk położony był na prezentację i nauczanie konkretnych architektur komputerów (np. VAX 11/780) i systemów mikroprocesorowych (i8080, i8086).

Należy zauważyć, że na obecnym poziomie rozwoju technologicznego architektura komputerów i projektowanie systemów mikroprocesorowych to w rzeczywistości jedna dziedzina – niemal wszystkie współczesne komputery są budowane na bazie mikroprocesorów, które pod względem mocy obliczeniowej wielokrotnie przewyższają komputery klasy *mainframe* sprzed kilku lat. Nauczanie projektowania systemów mikroprocesorowych zyskało więc znacznie szerszy i ogólniejszy niż dotychczas wymiar.

Szczególnie istotne staje się w tym kontekście nauczanie zasad, koncepcji, mechanizmów i struktur a nie omawianie poszczególnych układów. Konkretne procesory są w nowej formule nauczania traktowane jako przykłady implementacji, a nie jako główny przedmiot zajęć. System laboratoryjny wykorzystywany w tak zorganizowanym procesie nauczania powinien w istotny sposób różnić się od systemów używanych dotychczas.

Przegląd znanych rozwiązań

W dostępnych publikacjach można znaleźć opisy sprzętu używanego do celów dydaktycznych [2] [3] [4]. Na ogół są to mikrokomputery jednopakietowe o sztywno określonej konfiguracji, niekiedy z możliwością rozbudowy konfiguracji podstawowej. Są tu wykorzystywane proste, klasyczne mikroprocesory 8-bitowe (np. MC6800, Z80, Intel 8085) oraz mikroprocesory 32-bitowe (MC68020, i386). Systemy te są na ogół wyposażone w klawiaturę szesnastkową i wyświetlacz, umożliwiające interakcję z wbudowanym programem monitora albo też mają one możliwość współpracy z terminalem lub komputerem sterującym przez łącze szeregowe RS232C. Podstawowe wady takich systemów z punktu widzenia zastosowań dydaktycznych, to brak możliwości ingerencji w strukturę połączeń systemu, jego zamknięta architektura oraz brak lub ograniczenie możliwości monitorowania pracy systemu na poziomie sprzętowym (praca krokowa, obserwacja stanu szyny).

Używany dotychczas w Instytucie Informatyki PW (IIPW) do celów dydaktycznych i konstrukcyjnych system laboratoryjny MSM [5] został w nim opracowany na początku lat osiemdziesiątych jako modułowy system mikroprocesorowy ogólnego zastosowania. Takie przeznaczenie systemu wymagało dokładnego zdefiniowania standardu szyny i przesądziło o funkcjonalnym i konstrukcyjnym przystosowaniu wszystkich pakietów do pracy w pełnej konfiguracji użytkowej komputera, wyposażonej w konsolę i stacje dysków elastycznych. Określenie zarówno funkcji, jak i struktury modułów systemu – słuszne z punktu widzenia budowy instalacji użytkowych – spowodowało znaczne ograniczenie jego walorów dydaktycznych.

Głównymi wadami systemu (w tym kontekście) są:
- jednoznaczne związanie standardu szyny z układami rodziny Intel 8080,
- stosunkowo duża złożoność poszczególnych modułów,
- rozbudowany system układów buforujących,
- brak możliwości ingerencji w strukturę połączeń układów tworzących system,
- ograniczenie szerokości szyny danych (8 bitów),
- ograniczenie wielkości przestrzeni adresowej (64 KB).

Wewnętrzna struktura pakietów jest praktycznie niezmienna, co w znacznym stopniu zawęża możliwości projektanta: zamiast tworzyć nową konfigurację sprzętową musi się on koncentrować na analizie budowy i działania istniejących pakietów. Projektowanie jest w zasadzie ograniczone do adaptacji gotowych rozwiązań układowych, najczęściej przez odpowiednie zaprogramowanie użytych sterowników, rzadziej – przez dołączanie dodatkowych układów. Z doświadczeń autorów wynika, że wykorzystanie systemu MSM do realizacji zajęć laboratoryjnych wymaga przesunięcia głównego tematu ćwiczeć z aspek-

tów sprzętowych na oprogramowanie sterowników wejścia-wyjścia, a uruchamianie systemu wiąże się – przede wszystkim – z usuwaniem błędów programowych. Ponieważ definicja szyny MSMBUS nie odpowiada żadnemu z rozpowszechnionych standardów, system ten nie wydaje się również obecnie atrakcyjnym środowiskiem dla prac konstrukcyjnych.

W ostatnich latach rozbudowano system MSM i zmodyfikowano program ćwiczeń laboratoryjnych. Pakiety z procesorem Intel 8080 zastąpiono pakietami z procesorem Z80; wprowadzono również ćwiczenia zapoznające studentów z procesorami Intel 8088, NEC V-20 oraz MC68008, co pozwoliło na przedstawienie wybranych cech nowocześniejszych konstrukcji procesorów (kolejka instrukcji, obsługa sytuacji wyjątkowych, różne zakończenia cyklu transmisji).

Poszukując systemu, który mógłby zastąpić używany system MSM przeprowadzono analizę rynku krajowego. Przegląd ofert wykazał brak systemów przeznaczonych do celów dydaktycznych lub prototypowych. Swego rodzaju standardem de facto jest system modułowy MSM produkowany przez firmę IMPOL (wobec konfliktu nazw, w dalszej części tekstu stosowane będą oznaczenia IIPW-MSM i IMPOL-MSM). Pakiety systemu mają format pojedynczej eurokarty 3U, a szyna systemu odpowiada standardowi BUSMAT II. Pakiety zgodne z szyną BUSMAT II są również produkowane przez firmy DIALOG, MODULEX i GURU.

Z uwagi na wysoki stopień integracji funkcjonalnej pakietów nauczanie prowadzone przy wykorzystaniu takiego systemu ma głównie charakter demonstracyjny, a nie problemowy. Pozostają więc w mocy zastrzeżenia kierowane pod adresem systemu IIPW-MSM. Wypada jednak zauważyć, że mimo tych niedogodności system IMPOL-MSM jest wykorzystywany w Instytucie Cybernetyki Technicznej Politechniki Wrocławskiej. Opracowano tam dodatkowe moduły przystosowujące system do celów dydaktycznych (np. prosty monitor szyny) i moduły obiektowe, rozszerzające możliwości współpracy systemu ze światem zewnętrznym.

Koncepcja systemu

System laboratoryjny ma służyć do konstruowania prototypów systemów 8-, 16- i 32-bitowych, w tym również systemów wieloprocesorowych. Klasa mikroprocesorów możliwych do wykorzystania w systemie zawiera procesory uniwersalne oraz mikrokomputery jednoukładowe; przewidziana jest również możliwość użycia procesorów specjalizowanych (jak np. procesory sygnałowe, procesory graficzne, transputery). Budowa prototypów powinna odbywać się na poziomie połączeń międzyukładowych, co pozwala na konstruowanie systemów wykorzystujących układy (w tym mikroprocesory) różnych rodzin i praktyczne rozwiązywanie problemów z tym związanych. Wielkość budowanych konfiguracji jest ograniczona do kilku układów VLSI, tzn. pod względem złożoności odpowiada systemom jednopakietowym. Wskazane jest, aby poszczególne moduły systemu pełniły tylko jedną funkcję (np. procesor, port równoległy, pamięć). W przeciwieństwie do systemów IIPW-MSM i IMPOL-MSM nie jest przewidywane zastosowanie projektowanego systemu do budowy trwałych instalacji użytkowych.

Dla uzyskania założonej elastyczności konfiguracji systemu niezbędne jest zastosowanie konstrukcji modułowej. Ponieważ jednocześnie zakłada się możliwość ingerowania w strukturę połączeń na poziomie układów, stopień integracji poszczególnych modułów powinien odpowiadać poziomowi pojedynczych układów VLSI. Dla zagwarantowania pewności połączeń i odporności mechanicznej systemu pożądane jest zastosowanie klasycznej struktury szynowej. Standard szyny musi zapewniać możliwość konstruowania systemów mikroprocesorowych zawierających układy różnych rodzin.

Aby spełnić przedstawione wymagania szyna systemu powinna być pod względem standardu logicznego szyną poziomu międzyukładowego, tj. jej struktura logiczna powinna być zbliżona do struktury logicznej szyny zastosowanych układów mikroprocesorowych. Założenie możliwości budowania konfiguracji z mikroprocesorami różnych typów nie pozwala na ścisłe zdefiniowanie standardu logicznego szyny (ze względu na odmienną strukturę sygnałów sterujących transmisją w poszczególnych rodzinach układów). Rozwiązaniem jest przyjęcie „słabej definicji" szyny, która nie określając ściśle funkcji wszystkich sygnałów na szynie pozwala na trwałe przyporządkowanie i poprowadzenie stałych połączeń (np. linii zasilania). Pełna definicja dotyczy jedynie mechanicznego i elektrycznego standardu szyny, natomiast standard logiczny nie jest określony. W połączeniu z elastycznością połączeń w większości modułów systemu umożliwia to dostosowanie szyny do struktury sygnałów sterujących dowolnego mikroprocesora użytego w charakterze jednostki centralnej.

Pod względem struktury logicznej w systemie można wyróżnić trzy klasy modułów:
- moduły czynne, sterujące przebiegiem transmisji na szynie (moduły procesorów i sterowniki DMA),
- moduły bierne, odpowiadające na cykle transmisji inicjowane przez moduły czynne,
- moduły diagnostyczne, o szczególnych uprawnieniach w systemie, zdolne do monitorowania aktywności szyny i sterowania pracą krokową modułów czynnych.

W celu umożliwienia dopasowania modułów biernych do różnych standardów szyny struktura połączeń na tych modułach jest konfigurowana przez użytkownika. Moduły systemu zawierają zestaw układów (lub tylko podstawek) niezbędnych dla realizacji danej funkcji, ale większość z nich nie jest trwale połączona ze sobą. Należy zwrócić uwagę, że przy takiej koncepcji systemu poszczególne moduły są bardzo proste w budowie i praktycznie nie wymagają uruchamiania w fazie montażu. Uruchamianie modułu następuje każdorazowo po zdefiniowaniu struktury połączeń wewnątrz modułu przez użytkownika. W oczywisty sposób koresponduje to z głównym celem zajęć laboratoryjnych: nauczaniem projektowania i uruchamiania systemów mikroprocesorowych.

Mały stopień integracji poszczególnych modułów wraz z wymaganą dużą liczbą sygnałów szyny systemowej (niezbędnych w przypadku systemów 32-bitowych) implikuje wybór najmniejszego z typowych rozmiarów pakietów – pojedynczej eurokarty 3U (160 × 100 mm) wyposażonej w złącze pośrednie 96-stykowe. Kaseta systemu ma wbudowany zasilacz i miejsce na 10 pakietów.

Poszczególne moduły zawierają po jednym układzie VLSI wraz z podstawkami pod układy stanowiące jego niezbędną obudowę (jak np. dekodery, układy współpracy z szyną). Jedynie połączenia oczywiste z punktu widzenia funkcjonowania systemu są poprowadzone w postaci ścieżek obwodu drukowanego. Do takich połączeń należą przede wszystkim linie zasilania i danych, a w pewnych wypadkach – także linie adresowe. Wszystkie pozostałe połączenia, w tym połączenia sygnałów sterujących i synchronizujących, są realizowane przez użytkownika w trakcie uruchamiania prototypu przez łączenie układów za pomocą specjalnych przewodów z mikrowtyczkami wkładanymi w gniazda umieszczone na pakietach w rastrze 2.5 mm. Wszystkie nie podłączone na stałe wyprowadzenia układów są zaopatrzone w gniazda i – o ile to możliwe – jednoznacz-

nie opisane na pakiecie. Oprócz podstawek standardowych układów niezbędnych dla sprzęgania układów VLSI z szyną, na pakietach są umieszczane dodatkowe podstawki umożliwiające rozbudowę modułów.

Szyna systemu

Liczba połączeń dostępnych na łączówce kasety – po odliczeniu linii zasilań (14) – jest wystarczająca dla implementacji prostego systemu z 32-bitową szyną danych i 32-bitową szyną adresową, o nieskomplikowanej strukturze sterowania, lub systemów o mniejszych szerokościach szyn adresowych i danych z odpowiednio bogatszym repertuarem sygnałów sterujących.

Ściśle zdefiniowano jedynie niewielką liczbę sygnałów i własności funkcjonalnych systemu. Głównym powodem wprowadzenia jakiejkolwiek definicji jest uproszczenie struktury połączeń i ograniczenie liczby gniazd i długości połączeń na poszczególnych pakietach systemu. W miarę doraźnych potrzeb każda niewykorzystana zgodnie z pierwotną definicją linia szyny może być użyta do innych celów.

Standard szyny opisuje następujące cechy systemu:
- linie zasilania (14 linii) +5 V (6 linii), +12 V, −12 V, GND (6 linii) – w pełni określone,
- sygnały diagnostyczne (3 linie) – określone nazwy, znaczenie i położenie na szynie oraz zasady działania funkcji diagnostycznych w systemie,
- szyna adresowa (24 linie trójstanowe) – określone położenie linii na łączówce systemu,
- szyna danych (32 linie trójstanowe) – określone położenie linii na łączówce systemu,
- linie sterujące transmisją (8 linii trójstanowych) – określone położenie grupy linii na szynie, bez definicji znaczenia poszczególnych linii,
- linie systemu przerwań, arbitrażu szyny itp. (8 linii typu otwarty kolektor) – określone położenie grupy linii na szynie, bez definicji znaczenia poszczególnych linii,
- linie o połączeniu łańcuchowym (dwie linie – wejście i wyjście) – określone położenie linii na szynie, bez definicji znaczenia i zastosowania poszczególnych linii,
- linia przebiegu synchronizującego (zegara) – określone położenie, wykorzystanie pozostawione do dyspozycji użytkownika,
- linie definiowane przez użytkownika (4 linie) – określone położenie na szynie.

Na szynie systemu przewidziane są trzy linie sygnałowe o charakterze diagnostycznym:
- linia inicjacji systemu *–RESET* (typu otwarty kolektor, aktywna poziomem niskim),
- linia włączenia układu pracy krokowej *–SMODE* (typu otwarty kolektor, aktywna poziomem niskim); moduły aktywne z zaimplementowaną funkcją pracy krokowej powinny reagować na uaktywnienie tej linii przejściem do trybu pracy krokowej,
- linia sterowania pracą krokową *–STEP* (typ otwarty kolektor, aktywna zboczem opadającym); jej uaktywnienie zezwala na zakończenie bieżącego cyklu transmisji.

Współpraca modułów diagnostycznych z modułami aktywnymi jest jedyną silnie zdefiniowaną cechą protokołu szyny systemu (chociaż implementacja tej cechy przez użytkownika nie jest konieczna). Nie przewiduje się buforowania szyny na wszystkich pakietach wchodzących w skład systemu, lecz jedynie na pakietach modułów czynnych, definiujących standard logiczny szyny. Oznacza to, że konstrukcja poszczególnych pakietów będzie znacznie uproszczona w porównaniu z systemami IIPW-MSM i IMPOL-MSM, zatem także dokumentacja systemu będzie prostsza i bardziej zwarta.

Zasady konstrukcji modułów

Przyjęto, że moduł czynny powinien jak najdokładniej odpowiadać pod względem logicznym układowi lub grupie układów tworzących procesor systemu. Zestaw sygnałów na szynie modułu czynnego odpowiada zestawowi sygnałów procesora i określa standard logiczny szyny systemu. Możliwości konfiguracji modułów czynnych mogą więc być mniejsze niż w przypadku modułów biernych.

Dla umożliwienia implementacji podstawowych funkcji diagnostycznych jest wskazane wyposażenie modułów aktywnych w bufory pozwalające na asynchroniczne odłączenie procesora od szyny. Zastosowanie buforów zmniejsza ryzyko uszkodzenia układów tworzących moduł czynny – zwykle najdroższych z układów w danej konfiguracji systemu. Jeżeli procesor wykorzystany w module jest wyposażony w możliwość pracy krokowej, wskazana jest implementacja układu pracy krokowej sterowanego przez linie *–SMODE* i *–STEP* szyny systemu.

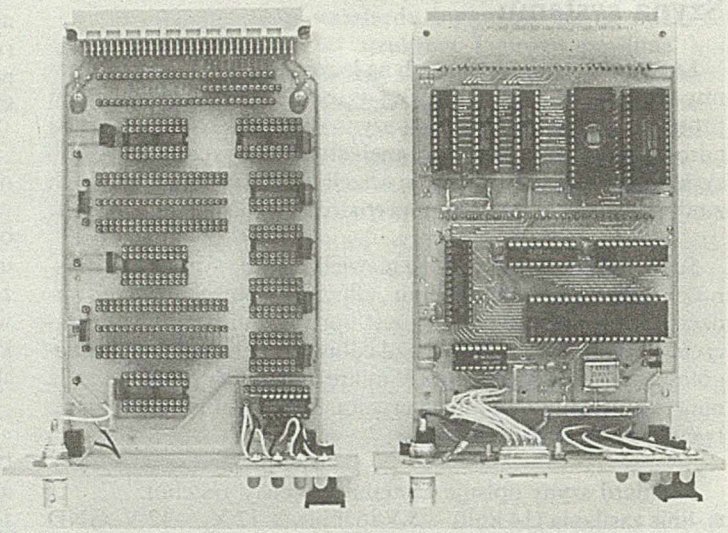
Moduł bierny, o ile nie jest dedykowany dla konkretnego typu procesora, powinien być skonstruowany w taki sposób, aby mógł współpracować z dowolnym modułem czynnym. Jednocześnie należy dążyć do uproszczenia procesu konfigurowania modułu biernego do pracy z wybranym procesorem. Ważne jest również zapewnienie dużej elastyczności konfiguracji, podwyższającej walory dydaktyczne modułu i pozwalającej na wykorzystanie go na wiele sposobów (np. możliwość synchronizaji z procesorem przez przerwania, testowanie stanu i wydłużanie cyklu transmisji). Złożoność modułu biernego powinna odpowiadać pojedynczemu blokowi funkcjonalnemu mikrokomputera, np. systemowi pamięci EPROM, portowi równoległemu, układowi zegara-licznika, sterownikowi wyświetlacza CRT. Zakłada się, że połączenia z szyną systemu nie są buforowane o ile nie jest to niezbędne. Buforowanie połączeń zewnętrznych zależy od decyzji użytkownika modułu.

W celu ułatwienia użytkowania systemu konstrukcja pakietów została do pewnego stopnia ujednolicona. Przykładowo: rozmieszczenie łączówek udostępniających sygnały szyny jest jednakowe dla wszystkich dotychczas zaprojektowanych modułów czynnych i biernych. Wszystkie moduły bierne zawierają ponadto identyczny zestaw podstawek, umożliwiających budowę dekodera adresów i układów synchronizacji transmisji z wykorzystaniem typowych elementów TTL (dekoderów, komparatorów, rejestrów, itp.) lub przy użyciu programowanych struktur logicznych.

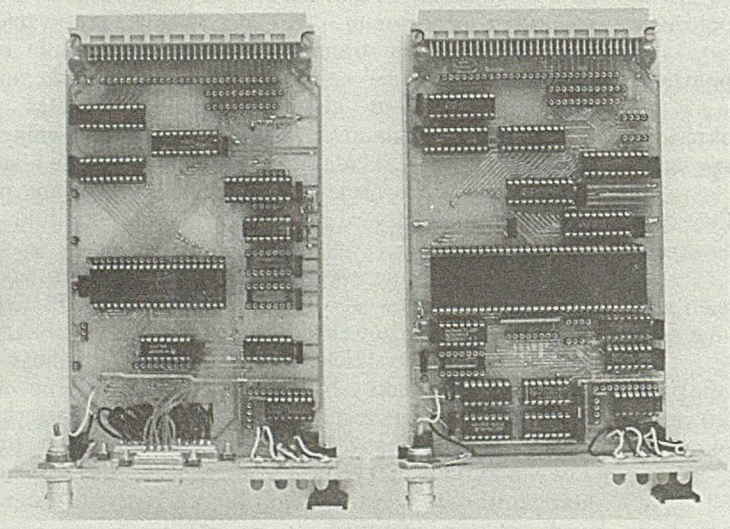
Każdy pakiet ma listwę maskującą, ułatwiającą wkładanie i wyjmowanie pakietu z kasety. Na niej są zamontowane diody elektroluminescencyjne, pozwalające na obserwację stanu wybranych sygnałów pakietu oraz gniazdo typu typu BNC, umożliwiające przyłączenie urządzenia monitorującego lub wytwarzającego sygnały (np. oscyloskopu, generatora) do dowolnego punktu na pakiecie. W zależności od funkcji pakietu, na listwie maskującej mogą być również umieszczone inne elementy, np. gniazda szufladowe lub wyświetlacze.

Przy projektowaniu dotychczas wykonanych modułów przyjęto, że na stałe poprowadzone będą jedynie połączenia konieczne w sposób oczywisty (np. linie zasilania). Taka konstrukcja modułów daje użytkownikowi swobodę w konfigurowaniu połączeń, dzięki czemu proces projektowania i uruchamiania systemu odbywa się na poziomie układów. Prosta organizacja modułów sprawia, że podstawowym dokumentem używanym przez studenta jest karta katalogowa wykorzystywanego procesora czy sterownika, a nie – jak ma to miejsce w przypadku innych podobnych systemów – opis rozwiązań zastosowanych przez konstruktora pakietu. W połączeniu z dowolnością

Najistotniejsze cechy systemu DSM – to rekonfigurowalny system połączeń i definiowalna struktura szyny.



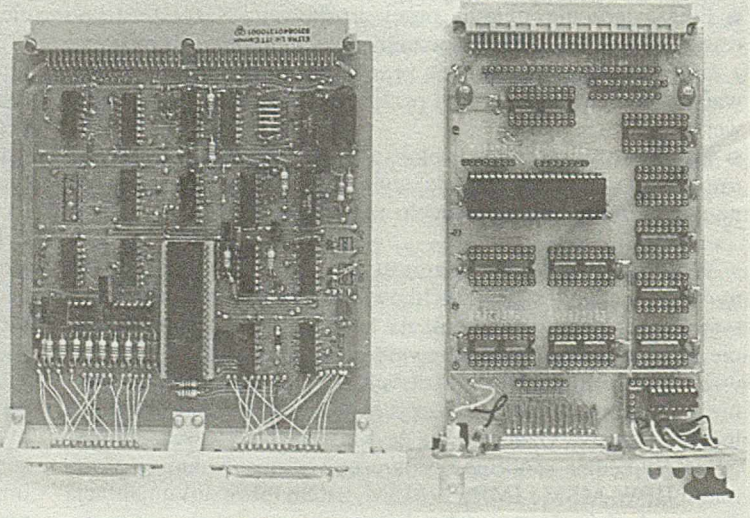
Na zdjęciu:
Pakiet prototypowy, który umożliwia dołączenie do systemu dowolnego układu i pakiet monitora szyny, współpracujący z komputerem nadrzędnym



Rezygnacja ze ścisłej definicji standardu szyny umożliwia użycie w systemie DSM dowolnych mikroprocesorów przy zachowaniu specyfiki sygnałów sterujących – wystarczy opracować nowy moduł i skompilować odpowiadający mu opis szyny.

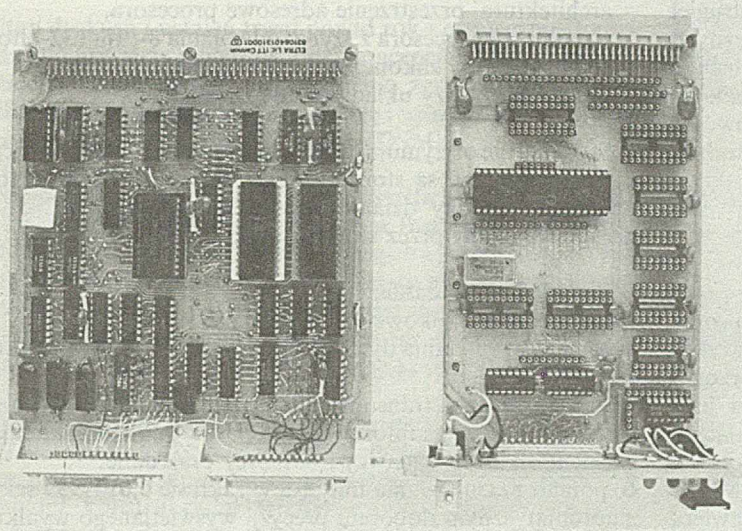
Na zdjęciu:
Pakiet procesora Intel 8051 i pakiet procesora Motorola 68000

W porównaniu z systemem IIPW-MSM, pakiety DSM mają mniejsze wymiary i prostszą strukturę. Znaczna część połączeń jest wykonywana przez użytkownika. Użytkownik określa także sposób buforowania połączeń zewnętrznych pakietu.



Na zdjęciu:
Pakiety równoległego wejścia-wyjścia z układami 8255 (IIPW-MSM i DSM)

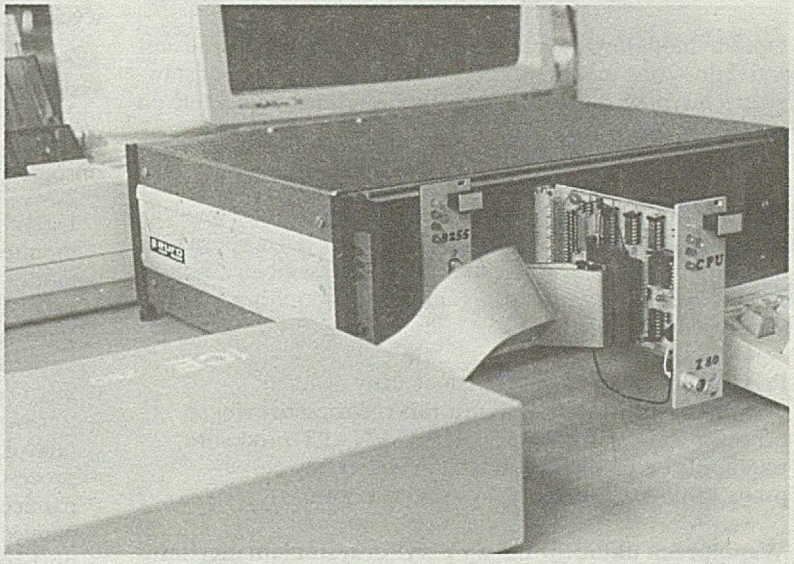
ten system jest lepszy?



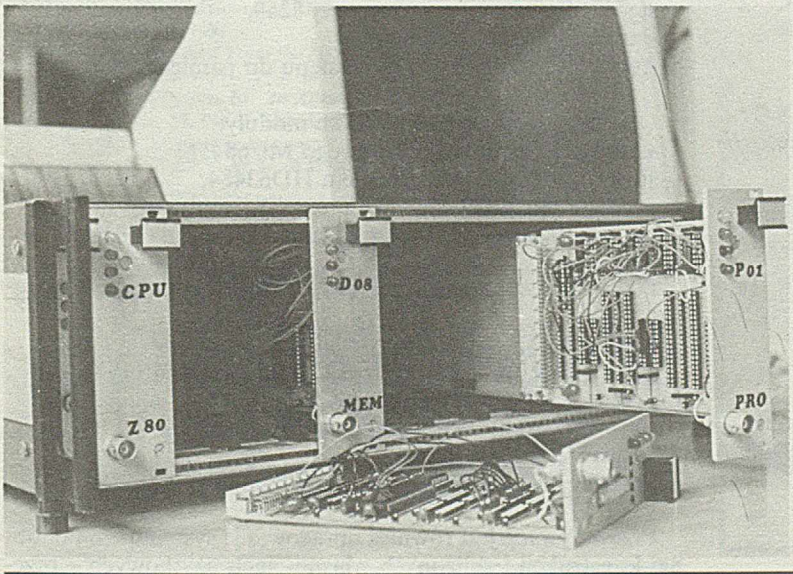
W odróżnieniu od innych systemów złożoność pakietów DSM odpowiada pojedynczym układom VLSI.

Na zdjęciu:
Pakiety transmisji szeregowej – IIPW-MSM (dwa sterowniki 8251A, układ zegarowy 8253, liczne układy towarzyszące) i DSM (sterownik 8250 i bufory linii ±12 V)

Wszystkie pakiety DSM są konstrukcyjnie przystosowane do uruchamiania.



Na zdjęciu:
Konfiguracja prototypowa z dołączonym emulatorem układowym mikroprocesora Z80 – widoczne diody LED i gniazda BNC na listwach maskujących; wszystkie układy są umieszczone w podstawkach



Konstruowanie prototypów w systemie DSM odbywa się na poziomie połączeń międzyukładowych.

Na zdjęciu:
Konfiguracja prototypowa podczas uruchamiania – widoczny pakiet na przedłużaczu szyny i kable połączeniowe

wyboru układów towarzyszących zbliża to znacznie przebieg zajęć dydaktycznych do rzeczywistego procesu projektowania.

Oczywiście pełna dowolność łączenia elementów przyczynia się do wzrostu częstotliwości uszkodzeń systemu, z drugiej jednak strony umożliwia przeniesienie zadań diagnostycznych z etatowego personelu laboratorium na użytkownika. Obsługa laboratorium ulega uproszczeniu dzięki eliminacji prac polegających na wyszukiwaniu uszkodzeń i naprawie pakietów. Ze względu na podwyższone prawdopodobieństwo uszkodzeń wszystkie układy są umieszczone w podstawkach.

Organizacja zajęć laboratoryjnych

Zestaw laboratoryjny składa się z kasety systemu oraz nadrzędnego komputera klasy IBM PC. Komputer jest wykorzystywany do przygotowania oprogramowania dla tworzonego systemu (edycja, translacja) oraz do wspomagania jego uruchamiania. Podstawowym narzędziem wspomagającym uruchamianie jest moduł monitora szyny umieszczony w kasecie systemu laboratoryjnego i sterowany z komputera IBM PC przez łącze szeregowe [1]. Pakiet monitora systemu zawiera mikrokomputer jednoukładowy i8051. Sterowanie pakietem monitora, przez oprogramowanie komputera nadrzędnego, odbywa się na podstawie definicji szyny systemu tworzonych w postaci tekstowej i kompilowanych przez kompilator opisów.

Oprócz modułów ze standardowymi blokami systemu jest również dostępny moduł prototypowy wyposażony w podstawki różnych typów, umożliwiający zmontowanie i dołączenie do systemu dowolnego układu.

W laboratorium dydaktycznym znajdują się trzy stanowiska uruchomieniowe. Każde stanowisko jest przeznaczone dla zespołu 2–3 osobowego. Komputer sterujący jest wyposażony w niezbędne oprogramowanie narzędziowe – asemblery, konsolidatory i symulatory oraz program obsługi monitora systemu laboratoryjnego.

Przyjęto następujący zakres tematyczny ćwiczeń:

1. Jednostka centralna systemu mikroprocesorowego:
- procesor Z80, stany WAIT, HOLD i HALT procesora,
- współpraca z pamięcią i układami we-wy,
- praca krokowa procesora, system przerwań.

2. Współpraca układów we-wy z wykorzystaniem mechanizmu DMA:
- sprzęganie pamięci i układów we-wy z szyną systemu,
- sterownik równoległego we-wy,
- sterownik transmisji DMA,
- realizacja przesłań DMA, transmisja blokowa.

3. Uzależnienia czasowe w systemach mikroprocesorowych:
- zasady obsługi typowych urządzeń zewnętrznych,
- układ licznika-zegara,
- sterownik przerwań,
- projekt systemu mikroprocesorowego.

4. Transmisja danych:
- transmisja szeregowa, łącze RS232C,
- sterownik transmisji szeregowej,
- synchroniczna i asynchroniczna współpraca z procesorem,
- projekt systemu mikroprocesorowego.

5. Procesor Intel 8051:
- architektura,
- współpraca z pamięcią i układami we-wy,
- stany procesora, realizacja specyficznych instrukcji,
- różnice pomiędzy układami Z80 a i8051.

6. Procesor Motorola 68000:
- architektura, przestrzenie adresowe procesora,
- współpraca procesora z szyną, odwołania 8-, 16- i 32-bitowe,
- różne warianty zakończenia cyklu dostępu,
- różnice pomiędzy układami Z80 a 68000.

Studenci nie otrzymują instrukcji do poszczególnych ćwiczeń, a tematy ćwiczeń są sformułowane jako zadania projektowe. Program sterujący działaniem systemu (ćwiczenia 2–5) jest przygotowywany przez studentów poza terminami zajęć.

Przykładowe zadania:
- układ pracy krokowej procesora Z80,
- układ wyświetlania dynamicznego wykorzystujący transmisję danych DMA,
- asynchroniczna transmisja szeregowa między trzema stanowiskami; przesyłana informacja ma postać pakietu zawierającego identyfikatory nadawcy i odbiorcy oraz dane,
- pomiar czasu trwania impulsu w zakresie 0,01–9999 sekund z automatycznym doborem precyzji wyświetlanego wyniku,
- generator symetrycznej fali prostokątnej; częstotliwość przebiegu wprowadzana z klawiatury,
- praca krokowa procesora MC68000 z wykorzystaniem różnych metod zakończenia cyklu szyny; obsługa sytuacji wyjątkowych „błąd szyny" i „nielegalna instrukcja".

Zadania są realizowane z wykorzystaniem dokumentacji systemu i oryginalnych opisów katalogowych poszczególnych układów, dzięki czemu studenci wykonują samodzielnie niemal kompletny projekt systemu na poziomie logicznym, a następnie oprogramowują i uruchamiają zaprojektowany sprzęt.

Dotychczas wykonano następujące moduły systemu:
- monitor szyny współpracujący z komputerem IBM PC,
- moduł prototypowy,
- procesor Z80 CPU,
- sterownik bezpośredniego dostępu do pamięci Z80 DMA,
- port równoległy Z80 PIO,
- zegar-licznik Z80 CTC,
- sterownik transmisji szeregowej Z80 DART,
- mikrokomputer jednoukładowy i8051,
- procesor MC68000,
- pamięć SRAM/EPROM z 8-bitową szyną danych,
- pamięć SRAM/EPROM z 16-bitową szyną danych,
- port równoległy 8255,
- zegar-licznik 8254,
- sterownik transmisji szeregowej 8250,
- sterownik przerwań 8259A,
- sterownik bezpośredniego dostępu do pamięci 8237.

Obecnie projektowane są nowe moduły:
- sterownik monitora ekranowego MC6845E,
- sterownik graficzny z układem HD63484,
- procesor MC68030,
- pamięć SRAM/EPROM z 32-bitową szyną danych.

Planowane jest wykonanie dodatkowych modułów, zawierających:
- wyświetlacz 7-segmentowy LED i przełączniki,
- wyświetlacz alfanumeryczny LCD i port klawiatury IBM PC,
- procesor sygnałowy TMS320C25,
- sterownik transmisji szeregowej MC2681,
- konwertery A/C i C/A,
- procesor 80C88 (możliwość zatrzymania zegara).

Przewidywane jest również opracowanie systemu – o podobnej konstrukcji i zbliżonych własnościach użytkowych – prze-

znaczonego do prac laboratoryjnych z zakresu sprzętu grafiki komputerowej.

Podsumowanie

Etap dyskusji i ustaleń trwał ponad rok i ostatecznie doprowadził do przyjęcia przedstawionej koncepcji w jej pierwotnym kształcie. Przemyślenia i spostrzeżenia zgromadzone w tym czasie znacznie przyspieszyły realizację i wdrożenie Dydaktycznego Systemu Mikroprocesorowego (DSM).

Począwszy od października 1991 r. system DSM w podstawowej konfiguracji jest wykorzystywany w Instytucie Informatyki PW do prowadzenia zajęć laboratoryjnych z przedmiotu „Technika Mikroprocesorowa". Wobec elastyczności i otwartości systemu rozważana jest również modyfikacja zajęć projektowych z tego przedmiotu, polegająca na doprowadzeniu do praktycznej realizacji prototypów projektowanych urządzeń.

Pod względem walorów dydaktycznych DSM przewyższa podobne systemy używane w uczelniach krajowych i zagranicznych (opisane w dostępnych publikacjach). W porównaniu z innymi konstrukcjami tej klasy system DSM ma wiele oryginalnych cech, podnoszących jego atrakcyjność w zastosowaniach dydaktycznych. Najważniejsze z nich to:

- niepełna definicja standardu szyny,
- skrajne uproszczenie logicznej struktury modułów,
- niewielka liczba ustalonych połączeń między układami,
- konstrukcyjne przystosowanie pakietów do uruchamiania.

DSM wzbudza zainteresowanie innych Instytutów i Uczelni; rozbudowana wersja systemu została wykonana dla Politechniki Białostockiej.

System został zgłoszony do nagrody Rektora Politechniki Warszawskiej za osiągnięcia dydaktyczne.

Dziękujemy wszystkim tym, którzy nam przeszkadzali, ponieważ ich upór utwierdzał nas w przekonaniu o konieczności wprowadzenia zmian. Dziękujemy także tym, którzy nam nie przeszkadzali – i bez tego mieliśmy dość kłopotów. Szczególne podziękowania należą się jednak tym, bez których nie doszłoby do zrealizowania DSM w jego obecnej postaci. **Zbigniew Kalisiak** i **Artur Chojnacki** wykonali monitor szyny wraz z pełnym oprogramowaniem; **Henryk Jeż** weryfikował powstające projekty i czuwał nad spójnością dokumentacji; **Romuald Siwik** i **Krzysztof Wacławski** wykonywali prace związane z montażem kaset i pakietów; wreszcie kierownik zespołu – **Marek Pawłowski** organizował finansowanie przedsięwzięcia i koordynował przebieg prac.

LITERATURA

[1] Chojnacki A., Kalisiak Z.: *Uniwersalny monitor szyny dla dydaktycznego systemu mikroprocesorowego*. Praca dyplomowa, II PW, 1992
[2] Fulcher J.: *Fun and Games and Microcomputer Interfacing*. IEEE Micro, Vol. 11, No. 1, February 1991
[3] Nicoud J.: *Dedicated Tools for Microprocessor Education*. IEEE Micro, Vol. 11, No. 1, February 1991
[4] Pollard L.H., Jordan R.: *An Advanced Educational Microprocessor System*. IEEE Micro, Vol. 11, No. 1, February 1991
[5] Woźniak A., Rzeszut J., Sosnowski J.: *Modularny System Mikroprocesorowy*. Raport Badawczy II PW nr 34, Warszawa 1981.

Przypominamy!

Jedyną możliwością regularnego otrzymywania INFORMATYKI jest prenumerata.

Czasopismo można zaprenumerować w dowolnym czasie i na dowolny okres w Dziale Kolportażu Wydawnictwa SIGMA-NOT Spółka z o.o. **00-716 Warszawa, ul. Bartycka 20, skr. poczt. 1004, telefony: 40-30-86, 40-35-89, 40 lub 40-00-21 w. 293, 295, 299.**

Zaległe numery można zawsze kupić **w redakcji tel. 39-14-34** lub **Dziale Handlowym: 00-950 Warszawa, skr. poczt. 1004, tel. 40-37-31.**

TERA Spółka Z O. O.

Przedsiębiorstwo Popierania Postępu TERA Spółka z o.o. uprzejmie informuje, że posiadając kilkuletnie doświadczenie w instalacji systemów wspomagających zarządzanie

o f e r u j e :

- opracowanie projektów systemów,
- optymalny kosztowo i rozwojowo dobór sprzętu i oprogramowania,
- instalację systemu u klienta,
- bezpłatnie przez rok konserwację oprogramowania oraz serwis sprzętu wraz z doradztwem techniczno-eksploatacyjnym,
- szybką dostawę uzupełnień konfiguracji lub sprzętu mikrokomputerowego niezależnego od systemu, w tym mikrokomputerów ALR firmy Wearnes Technology.

Wszelkie dodatkowe informacje uzyskają Państwo codziennie oprócz niedziel w Biurze Handlowym

40-025 Katowice, ul. Wojewódzka 31
tel. (faks): 155-26-72, teleks: 315448 tera pl

PAMIĘTAJ! Instalacje XENIX/NOVELL/PC MOS 386 oraz serwis to nasza specjalność – ZAPRASZAMY, ponieważ czterech lat doświadczeń nigdzie nie kupisz.

0/19/90

meditronik SPÓŁKA z o.o.

- CZĘŚCI ELEKTRONICZNE
- KOMPUTERY PS/1,PS/2
- DRUKARKI HP
- INSTALACJE SIECI KOMPUTEROWYCH

Partnerzy handlowi: ANALOG DEVICES, ITT, MOTOROLA, SAMSUNG, TELEFUNKEN i inni

UMC UNITED MICROELECTRONICS CORPORATION

- UKŁADY PAMIĘCI
- UKŁADY KOMPUTEROWE
- UKŁADY KOMUNIKACYJNE I KOMERCYJNE

PRZEDSTAWICIELSTWO

HEWLETT PACKARD COMPONENTS

- TRANSOPTORY
- WSKAŹNIKI ŚWIETLNE
- WYŚWIETLACZE LED
- PRODUKTY KODÓW KRESKOWYCH
- KONTROLERY I CZUJNIKI RUCHU
- TECHNIKA ŚWIATŁOWODOWA
- ELEMENTY W.CZ. I MIKROFALOWE
- PODZESPOŁY DO MONTAŻU POWIERZCHNIOWEGO (SMD)

DYSTRYBUCJA

BOURNS

- POTENCJOMETRY TRIMPOT
- HYBRYDY REZYSTOROWE
- REZYSTORY SUBMINIATUROWE
- BEZPIECZNIKI MULTIFUSE
- POTENCJOMETRY PRECYZYJNE
- POTENCJOMETRY PANELI CZOŁOWYCH I KODERY
- CEWKI I TRANSFORMATORY
- CZUJNIKI CIŚNIENIA, POŁOŻENIA I PRZYŚPIESZENIA

PMI

DYSTRYBUCJA

IBM Business Partner

meditronik sp. z o.o.

00-194 Warszawa,ul. Dzika 4
tel. (02) 6352263, 6352264
fax (02) 6352195, tlx 816075

SO/375/91

ParaSoft

Dystrybutor firm : Lahey Computer Systems, Inc.
BORLAND International Inc.

Kiedy ludzie mówią o FORTRANIE, najczęściej wymienianą nazwą jest Lahey

Computer Systems, Inc.

F77L-EM/32 & Lahey Ergo OS/386

Przy pomocy tego kompilatora można przenieść programy z dużych maszyn na PC 386/486 Jest to 32-bitowy kompilator, którego nowa wersja 4.0 zawiera między innymi: rozbudowane narzędzia programisty, obsługę wirtualnej pamięci, programowanie w modzie chronionym, współpracę z DESQview, nową dokumentację.

F77L

Bardzo szybki kompilator pracujący w trybie rzeczywistym. Kompilator ten pozwala wykorzystać 32-bitowe instrukcje na komputerze 386. Nowa wersja 5.0 zawiera między innymi: obsługę kooprocesora Weitek, Video Grafikę.

Lahey Personal Fortran 77

Doskonałe narzędzie do nauki Fortranu. Wersja 3.0 zawiera pełny Fortran standardu ANSI 77, Edytor, Debugger, Linker, Microsoft i Borland C interface.

Borland C++ 3.0	2.340.000
Borland C++ 3.0 & Appl. Frameworks	3.900.000
Turbo C++ 2nd	850.000
Turbo C++ & TurboVision	1.320.000
Turbo C++ for Windows	1.320.000
Turbo Pascal 6.0	850.000
Turbo Pascal 6.0 Pro	1.320.000
Turbo Pascal for Windows	1.200.000
Turbo Debugger & Tools	1.380.000
ObjectVision 2.0	1.200.000
Paradox 3.5	2.400.000
Paradox Engine 2.0	1.200.000
Quattro Pro 3.0	1.560.000
Sprint	1.200.000
SideKick 2.0	920.000

Promocja !

BORLAND INTERNATIONAL

Application Frameworks	1.000.000
Turbo Vision	600.000
Quattro Pro SE + Paradox SE	1.000.000
dBase IV PL + Quattro Pro 3.0	3.000.000
Turbo Pascal for Windows + Turbo C++ for Windows	2.000.000

Do pakietów:
Pascal for Windows
C++ for Windows
dodajemy gratis programy:
SCREENERY lub Resource Workshop (do wyboru)

To nie koniec naszych propozycji !!!

☎ tel/fax (115) 210-94 ✍ 32-400 Myślenice; Słowackiego 17/9

Informujemy, że prowadzimy również sprzedaż wysyłkową

0/2/92

ZDZISŁAW WRZESZCZ
Warszawa

Inteligentny system gromadzenia informacji faktograficznej

Przedmiotem naszego zainteresowania jest metodologia gromadzenia i przetwarzania informacji faktograficznej (informacji FG). Elementami strktur faktograficzych są dane dotyczące przede wszystkim zachowań różnych obiektów tworzących sytuację problemową. Uwagę obserwatora sytuacji skupiają zachowania szczególne zwane zdarzeniami. Celem gromadzenia informacji FG jest dokumentowanie stanów sytuacji problemowych, w których miały miejsce określone zdarzenia. Dokumentacja taka może być podstawą w późniejszym poszukiwaniu zależności między zachowaniem obiektów i zdarzeniem. Rozpatrzmy na wstępie przykłady takich sytuacji.

Sytuacja 1. Prototyp pewnego nowego urządzenia, np. transportera górniczego, jest poddawany próbom w warunkach rzeczywistych. Projektant transportera uwzględnił w obliczeniach pewne warunki eksploatacji urządzenia, na które składało się m.in. przeciętne obciążenie taśmy, wymagane szybkości taśmy, temperatura otoczenia itp. Poddany badaniom w rzeczywistym procesie prototyp wykonuje cyklicznie wyznaczoną operację. Operacja ta jest wykonywana w większości przypadków prawidłowo, tj. zgodnie z założeniami projektanta urządzenia. Istnieją jednak przypadki nieprawidłowego wykonania operacji przez urządzenie. Przypadki te, to właśnie zdarzenia wymagające udokumentowania. Przyczyny tych zdarzeń można poszukiwać w zachowaniu tych elementów sytuacji (uczestniczących w procesie technologicznym), których zachowanie nie zostało w dostatecznym stopniu uwzględnione przez projektanta. Do elementów takich może być np. zaliczony ładowacz urobku podający zbyt wielkie bloki na taśmę, operator napędu, ruch podłoża zmieniający ustawienie urządzenia itp. Informacja faktograficzna powinna więc dotyczyć nie tylko zachowań elementów transportera, ale też i tych elementów sytuacji, które mogą być odpowiedzialne za wywołanie zdarzenia niesprawnej pracy transportera.

Sytuacja 2. Po drogach wybranego terenu poruszają się samochody transportu, samochody osobowe, wozy konne, zwierzęta, osoby. Ponadto, na terenie tym znajdują się budynki mieszkalne, gospodarskie i fabryczne. Wszystkie te obiekty z racji swojej działalności, wydalają do otoczenia gazy, wodę, żużel itp. zawierające szkodliwe substancje. Ilość i rodzaje substancji wydalanych przez obiekty do otoczenia powinna odpowiadać określonym normom. Jeśli te ilości przekraczają normy, to mamy do czynienia ze zdarzeniem zatrucia środowiska. Chcąc poznać sprawców, trzeba gromadzić informację faktograficzną.

Oprócz sytuacji związanych z przemysłem można też przytoczyć przykłady z różnych badań doświadczalnych w naukach podstawowych (fizyka, chemia, biologia), technicznych, medycznych, społecznych, a także z ochrony środowiska, rolnictwa, czy też wytwarzania żywności.

Gromadzenie informacji faktograficznej jest niezbędne wszędzie tam, gdzie zachodzi potrzeba nie tylko dokumentowania, lecz również badania przyczyny zdarzeń, a także ich prognozowania.

Cechą szczególną sytuacji problemowych, jak to pokazują przykłady, jest dominacja jakościowych charakterystyk w opisie tych sytuacji. Bardziej istotne jest to, że jakiś obiekt jest usytuowany na pewnym podłożu, niż to, że ten obiekt znajduje się w punkcie o wartościach współrzędnych w przestrzeni X, Y, Z.

Nowość zagadnienia gromadzenia informacji faktograficznej, złożoność tej informacji, a ponadto dominacja jakościowych form jej opisu powodują, że w pracy nad tym zagadnieniem [8, 9] sięgnięto do metod i środków sztucznej inteligencji (ang. *Artificial Intelligence* – AI) [3].

Niniejszy artykuł zawiera – uzyskane przez autora – wyniki dotyczące języka formułowania struktur informacji FG. Prezentuje też algorytmy gromadzenia tych struktur w postaci ciągu pierwotnego. Praca nad systemem CFG (Gromadzenie Faktografii) była rozpoczęta w Instytucie Maszyn Matematycznych w ramach problemu RPBR 3, cel nr 16: „Opracowanie metod gromadzenia, reprezentacji wiedzy i wnioskowania w systemach informacyjnych".

Koncepcja systemu

Zadaniem niniejszego rozdziału jest przedstawienie wybranego tu ujęcia informacji faktograficznej oraz modelu systemu jej gromadzenia.

Określenia podstawowe

Określenie **informacja faktograficzna** można spotkać wszędzie tam, gdzie mówi się o zapisie faktów, tj. danych nie budzących wątpliwości i dotyczących zdarzeń, właściwości obiektów itp. Zwróćmy uwagę, że połączenie prawdziwych faktów w formę pewnej struktury informacyjnej może sugerować istnienie nowych treści, np. zależności przyczynowych między zdarzeniem i pewnymi właściwościami obiektów. Obserwacja w licznych sytuacjach dowodzi, że ten właśnie rodzaj informacji faktograficznej budzi szczególnie zainteresowanie.

Obserwacji **zdarzeń** dokonujemy przez obserwację stanu wyróżnionych obiektów w danym obszarze obserwacji.

Obszarem **obserwacji** jest dowolny wyróżniony fragment świata rzeczywistego. Dowolny element obszaru obserwacji może stać się **obiektem** obserwacji, jeśli zdaniem inżyniera problemu obiekt ten może mieć związek ze zdarzeniem. Obiektem obserwacji jest również cały obszar obserwacji.

Każdy obiekt obserwacji może być określony przez swoje atrybuty. **Atrybutem** obiektu obserwacji jest dowolna właściwość obiektu obserwacji, uznana przez inżyniera problemu jako istotna. Atrybut obiektu może określać naturę obiektu, jego strukturę, kształty, rozmiary, zdolności funkcjonalne, zdolności oddziaływania na otoczenie itd.

Atrybuty obiektu obserwacji mogą mieć charakter stały bądź też zmienny w czasie. Atrybuty **stałe** są podstawą definicji obiektu, atrybuty **zmienne** w czasie określają zachowanie obiektu. Atrybuty zmienne służą więc do manifestacji zdarzenia

(atrybuty zmienne obiektu pasywnego) i do badania **zależności przyczynowej** między zachowaniem obiektów aktywnych a zdarzeniem.

Inżynierem problemu będziemy nazywać specjalistę formułującego problem gromadzenia informacji FG dla konkretnego obszaru obserwacji.

Zakładamy, że obserwacja zdarzenia będzie dokonywana przez obserwację atrybutów zachowania pewnego wyróżnionego obiektu. Obiekt ten można nazywać **pasywnym,** zaś pozostałe obiekty obserwacji – **aktywnymi,** przy założeniu, że mogą one oddziaływać na zdarzenie.

Opis **aktywności** obiektu obserwacji powinien uwzględniać pełen opis obiektu, t.j. atrybuty stałe oraz atrybuty zachowania.

Model systemu

Pakiet. Jest to podstawowa forma zapisu informacji faktograficznej. W pakiecie są zawarte aktywności obiektów z danego obszaru obserwacji w połączeniu z wartością czasu, w którym dokonano obserwacji oraz określeniami obiektów. Zapisy takie mają następującą formę:

⟨pakiet⟩ :: = ((⟨data obserwacji⟩ ⟨czas obserwacji⟩)
(⟨obiekt pasywny /**OP**/⟩ ⟨atr. zachow. **OP**⟩ [⟨atr. zachow. **OP**⟩])
(⟨obiekt aktywny/**OA**/⟩μ⟨atr. zachow. **OA**⟩μ[⟨atr. zachow. **OA**⟩])
[(⟨**OA**⟩ ⟨atr. zachow. **OA**⟩ [⟨atr. zachow. **OA**⟩])]

gdzie:
⟨w⟩ – dowolne wyrażenie należące do określonego języka symbolicznego zapisu obiektów,
[⟨w⟩] – zero- lub wielokrotne powtórzenie elementu ⟨w⟩.

Ciąg FG. Bardziej miarodajną formą zapisu informacji faktograficznej odnoszącą się do określonej sytuacji problemowej jest ciąg pakietów faktograficznych (ciąg FG):

⟨ciąg **FG** :: = ⟨pakiet⟩ [⟨pakiet⟩]

Hipoteza o zależności zdarzenia od zachowania obiektu (hipoteza HB). Hipotezę HB podajemy w postaci dwóch zbiorów, z których pierwszy:

PO-BATi-N = {**PO-BATi**:i = 1..k}

określa atrybuty zdarzenia, zaś drugi:

AO-BAT-N = {**AO-BATi**: i = 1..l}

określa atrybuty zachowania obiektów aktywnych.

Zgodnie z hipotezą HB dowolny obiekt aktywny ⟨AO⟩, którego zachowanie określa zbiór atrybutów {AO – BATi} ⊇ ⊇ AO – BAT – N, może przyczynić się do wywołania zdarzenia ⟨E⟩ określonego zbiorem atrybutów {PO – BATi} ⊇ ⊇ PO – BAT – N, jeśli co najmniej jeden atrybut zachowania ⟨AO – BATi⟩ należy do zbioru {AO – BATi}.

Hipoteza o łącznym pozytywnym wpływie zachowań obiektów aktywnych na zdarzenia (hipoteza HC). Jeśli w wyniku pewnej liczby obserwacji stwierdzono, że zdarzeniu E1 towarzyszą zachowania {AO–BAT1}...{AO–BATn} obiektów aktywnych O1...On, to wymienione zachowanie tych obiektów może być traktowane jako przyczyna zdarzenia E1 aż do chwili zaobserwowania zdarzenia E2 w następstwie tych samych zachowań wymienionych obiektów.

Komunikacja systemu z obszarem obserwacji. System komunikuje się z obszarem obserwacji za pomocą pewnego **pośrednika.** Do zadań pośrednika należy przede wszystkim obserwacja atrybutów określonych obiektów i wprowadzanie wyników tej obserwacji do systemu. Kolejne zadanie pośrednika jest formułowane przez system. System pełni więc rolę wiodącą w tym dialogu i w związku z tym musi dysponować stosowną do tej roli **wiedzą.** Pośrednikiem może być: inżynier systemu, użytkownik systemu lub automatyczne urządzenie pomiarowo-obserwacyjne, mające zdolność interpretowania języka poleceń systeu (patrz rozdział „Przykład działania systemu").

Działanie systemu. Celem działania systemu jest formułowanie i **gromadzenie** ciągów FG w **bazie danych** FG systemu oraz **wyszukiwanie** pakietów spełniających określone kryteria wyszukiwania. Czynności gromadzenia i wyszukiwania informacji FG dla nowej sytuacji problemowej powinny być poprzedzone czynnością **zakładania bazy wiedzy** o obiektach aktywnych i zdarzeniach.

Algorytm ogólny systemu (algorytm α). Zwróćmy uwagę, że implikowana treść pakietów danego ciągu FG jest silnie uzależniona od doboru obiektów aktywnych do obserwacji. Przyjmujemy następujący ogólny algorytm działania systemu:
(α 1) zgromadzenie w **bazie** FG (baza danych systemu) ciągu **pierwotnego,** t.j. ciągu FG zawierającego pakiety dotyczące wszystkich obiektów odnotowanych w obszarze obserwacji;
(α 2) utworzenie ciągu **wtórnego** poprzez **filtrację** ciągu pierwotnego.

Kryteria filtracji ciągu pierwotnego wynikają z określonej przez inżyniera problemu **heurystyki filtracji.** Elementami języka formułowania heurystyki filtracji są wszystkie elementy zapisu pakietu, a całość heurystyki jest pewnym odbiciem określonej hipotezy obserwacji. Algorytmy systemowe różnią się więc ze względu na heurystyki filtracji.

Algorytm β. Algorytmem β będziemy nazywać następującą sekwencję postępowania systemu:
(β 1) utworzenie repliki obszaru obserwacji w pamięci systemu;
(β 2) sformułowanie zbiorów PO–BAT–N oraz AO––BAT–N dla hipotezy HB;
(β 3) zgromadzenie pierwotnego ciągu FG dla danego zdarzenia; pakiety ciągu **pierwotnego** zawierają aktywności wszystkich obiektów, wprowadzone w wyniku dialogu systemu i pośrednika;
(β 4) utworzenie wtórnego ciągu FG dla danego zdarzenia; wtórny ciąg FG powstaje w wyniku filtracji ciągu pierwotnego na podstawie hipotezy HB;
(β 5) sformułowanie reguł kwalifikacji obiektów związanych przyczynowo ze zdarzeniem;
(β 6) zgromadzenie ciągu FG, w którym obiekty aktywne są zgodne z regułami kwalifikacji;
(β 7) weryfikacja reguł kwalifikacji.

Możliwe jest uformowanie i realizacja dowolnego innego algorytmu zawierającego kroki zakładania repliki obszaru obserwacji, obiektów, heurystyk filtracji, bazy wzorców zdarzeń itp., pod warunkiem, że kroki te znajdą odwzorowanie w środkach realizacyjnych systemu tworzących procesy systemu.

Reprezentacja wiedzy

System musi dysponować wiedzą niezbędną do automatycznego działania zgodnie z określonym algorytmem, a mianowicie wiedzą umożliwiającą:

- utworzenie w pamięci repliki obszaru obserwacji wraz z występującymi tam obiektami obserwacji,
- wprowadzenie heurystyk filtracji,
- utworzenie reguł kwalifikacji.

Schemat – postać ogólna. Do reprezentowania wiedzy o obszarze i obiektach obserwacji wybrano strukturę nazwaną tu **schematem.** Pierwowzorem schematu jest rama zaproponowana przez Minskyego do analizy scen [5]. Klatki ramy zawierały opisy elementów sceny.

Schemat jest zestawem klatek, zawierających atrybuty obiektu obserwacji. Klatka może też zawierać procedurę pomocną dla mechanizmu sterowania systemem. Spośród rozpatrywanych metod reprezentacji wiedzy rama, a w ślad za nią schemat, dysponują największą siłą ekspresji, a przez to umożliwiają tworzenie homeomorficznej **repliki sytuacji problemowej.** Homeomorfizm repliki jest niezbędny wtedy, gdy chcemy odtworzyć w systemie metody rozwiązywania problemu stosowane przez inżyniera problemu. A oto ogólna postać schematu:

(⟨symbol repliki⟩ (⟨atr⟩ ⟨asp⟩ [⟨inf⟩]))
[(⟨atr⟩ (⟨asp⟩ [⟨inf⟩]))])

Znaczenie elementów tego wyrażenia jest następujące:
⟨symbol repliki⟩ – dowolny element ze zbioru ⟨symbol repliki⟩ zawierającego symbole obszaru i obiektów obserwacji, atom:

⟨asp⟩ ::= WA | SUB | DEM

| – łącznik logiczny „lub";
WA – aspekt wartości, symbol stały;
SUB – aspekt zastępczy, symbol stały;
DEM – aspekt procedury, symbol stały;
⟨atr⟩ – symbol atrybutu, atom;
⟨inf⟩ – dowolny element ze zbioru wartości przyjmowanych przez atrybut danej klatki, s-wyrażenie;
[ ] – zero lub więcej powtórzeń zawartego wyrażenia.

Występujące w wyrażeniu nawiasy można traktować jako wyróżniki klatek zawierających poszczególne atrybuty; w istocie jednak schemat zamierzamy implementować jako s-wyrażenie języka symbolicznego Lisp [4].

Postać ogólna schematu posłuży nam do tworzenia schematów poszczególnych składników sytuacji problemowej.

Schemat obszaru obserwacji. Replikę obszaru obserwacji opiszemy następująco:

(⟨symbol obszaru obserwacji⟩ (**OPIS** (**VA** ⟨txt⟩))
(**KLASA** (**VA** ⟨symbol wzorca⟩))
(**ELEMENTY** (**VA** ⟨inf⟩ [⟨inf⟩])
[(⟨atr⟩ (⟨asp⟩ ⟨inf⟩ [⟨inf⟩]))])

Schemat obszaru ma więc trzy ustalone atrybuty oraz zero lub więcej atrybutów specyficznych dla danego problemu. Zawartość pola OPIS.txt jest przeznaczona dla programisty lub użytkownika systemu. Pole KLASA.inf zawiera symbol schematu wzorca (klasy) reprezentacji: atrybut ten jest związany z mechanizmem **dziedziczenia właściwości** przez schemat podporządkowany danej klasie. Wykorzystanie mechanizmu dziedziczenia ma ogromne znaczenie dla programowania każdego złożonego systemu: wprowadza przejrzystość do organizacji programu oraz usuwa redundację zapisów w pamięci systemu.

Schemat obiektu. Dowolny obiekt tworzący sytuację problemową jest rozwinięciem zapisu elementu podobszaru, rozwinięcie to ma postać następującego schematu:

(⟨symbol obiektu⟩ (**OPIS** (**VA** ⟨txt⟩))
(**KLASA** (**VA** ⟨symbol wzorca⟩))
(**ROLA** (**VA** ⟨r-inf⟩))
[(**ELEMENTY** (**VA** [⟨inf⟩]))]
(⟨bat-n⟩ (**VA** ⟨bat-v⟩))
[(⟨bat-n⟩ (**VA** ⟨bat-v⟩))]
[(⟨atr⟩ (⟨asp⟩ ⟨inf⟩ [⟨inf⟩]))])

gdzie:
bat-n – nazwa atrybutu zachowania,
bat-v – wartość atrybutu zachowania,
ROLA – atrybut określający rolę obiektu w systemie,
⟨r-inf⟩ ::= A | P,
A – aktywna rola obiektu,
P – pasywna rola obiektu.

Implementacja systemu

Implementacja systemu gromadzenia informacji faktograficznej będzie polegać na utworzeniu środków do formułowania repliki sytuacji problemowej oraz środków do realizacji ciągów FG zgodnie z algorytmem α. Do realizacji tych środków zastosujemy język Common LISP.

Common LISP [5] jest dialektem języka Lisp [4]. Podobnie jak oryginał, Common LISP nie dysponuje rozwiniętymi strukturami do formułowania podobnych problemów, stanowi on jednak niezwykle elastyczną bazę, umożliwiającą tworzenie systemów oryginalnych. Zalety języka i jego główne elementy były już wcześniej prezentowane na łamach INFORMATYKI [10].

Operacje schematu

W związku z wybraną metodą reprezentacji wiedzy, podstawowym składnikiem środków implementacji stają się operacje tworzenia schematu i operacje manipulacji zawartością klatek schematu. Przedstawimy najważniejsze z tych operacji.

Podstawowe znaczenie dla konstrukcji procedur operacyjnych ma decyzja o sposobie pamiętania struktur zwanych tu schematem. Wybrany sposób był już stosowany w zbliżonej postaci w pewnych konstrukcjach opartych na ramie. Sposób ten polega na pamiętaniu struktury schematu (bądź ramy) na liście własności obiektu ⟨symbol-obiektu⟩ jako jego własność np. **SCH** od nazwy schemat.

Pobranie zawartości atrybutu. Zawartość atrybutu dowolnego schematu można otrzymać jako wynik operacji:

(**WEZI** ⟨sch⟩ ⟨atr⟩ ⟨asp⟩)

gdzie:
⟨sch⟩ – symbol schematu,
⟨atr⟩ – nazwa atrybutu,
⟨asp⟩ ::= **VA** | **SUB** | **DEM**

Wprowadzenie zawartości atrybutu. Operacja:

(**WPROWI** ⟨sch⟩ ⟨atr⟩ ⟨asp⟩ ⟨inf⟩)

wprowadza daną ⟨inf⟩ w wyznaczonym aspekcie atrybutu danego schematu.

Usuwanie zawartości atrybutu. Operacja:

(**USUNI** ⟨sch⟩ ⟨atr⟩ ⟨asp⟩ ⟨inf⟩)

usuwa określoną treść wyznaczonego aspektu atrybutu danego schematu.

Wykorzystanie aspektu zastępczego. Powyższe operacje mogą być wspomagane przez operacje pobrania zawartości aspektu SUB w przypadku, gdy w aspekcie VA nie znajduje się nic:

(**WEZI-SUB** ⟨sch⟩ ⟨atr⟩)

Operacja ta umożliwia wykorzystanie mechanizmu dziedziczenia.

Procesy systemu

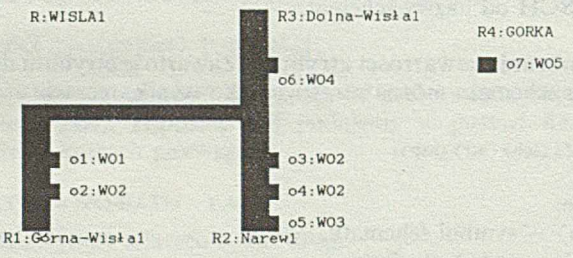
Działanie systemu zamyka się w obrębie oddzielnych kroków określonego algorytmu, np. algorytmu α. Realizacji tych kroków nadajemy charakter pewnego **procesu dialogowego,** wywołanego przez użytkownika systemu, lecz kierowanego – w trakcie jego realizacji – przez system. Po wywołaniu danego procesu rola użytkownika sprowadza się do roli dostawcy danych dla określonych atrybutów wyznaczonego obiektu. Z charakterystyką procesów dialogowych zapoznany się na przykładzie pewnej pozorowanej aplikacji.

Przykład działania systemu

Sformułowano pewną quasi-rzeczywistą sytuację problemową z dziedziny ochrony środowiska oraz program pod nazwą GFG (gromadzenie informacji faktograficznej) do gromadzenia informacji FG dotyczącej sytuacji danego typu.

Opis obszaru obserwacji i zadanie systemu

Niech obszarem obserwacji będzie pewien teren wokół rzeki pod nazwą WISŁA1, obejmujący główny nurt rzeki i jej dopływ. Wzdłuż brzegów rzeki i jej dopływów są umiejscowione zakłady przemysłowe, warsztaty naprawcze i gospodarstwa rolne. Wszystkie te obiekty pobierają wodę z rzeki, niezbędną do ich procesów technologicznych, a następnie, po wykorzystaniu w procesie technologicznym, zwracają tę wodę (zrzut). Jeśli przedsiębiorstwo nie stosuje filtracji do zrzutu wody (brak filtru, czasowe zdjęcie lub uszkodzenie filtru), to zawarte w zrzucie, a pochodzące z procesu technologicznego składniki, powodują zanieczyszczenie rzeki. W kategoriach tworzonego systemu jest to zdarzenie, natomiast przedsiębiorstwa zanieczyszczające teren, to obiekty aktywne.



Szkic obszaru obserwacji dla zadania przykładowego

Oznaczenia:
- obszary:
  - R – obszar obserwacji,
  - R1, R2, R3, R4 – podobszary obserwacji,
- obiekty:
  - 01, ..., 07 – obiekty obserwacji,
- wzorce obszarów:
  - WR1 – obszar rzeczny zabudowany,
  - WR2 – teren przybrzeżny,
  - WR3 – wzgórze,
- wzorce obiektów:
  - W01 – warsztat,
  - W02 – gospodarstwo rolne,
  - W03 – fabryka nawozów,
  - W04 – zbiornik kontrolny,
  - W05 – warunki atmosferyczne

Cel działania systemu. System GFG staje się narzędziem w programie obserwacyjnym, którego celem jest dostarczanie na bieżąco informacji o jakości wody obszaru rzecznego WISŁA1. Jakość wody jest badana w związku z jej określonym przeznaczeniem. Jakość ta jest mierzona za pomocą **parametrów jakości.** Dobór tych parametrów ma ogromne znaczenie dla całego programu obserwacji [7]. Mówiąc ogólnie, parametr jakości powinien dotyczyć pewnych ogólnych własności próbki, a nie właściwości pojedynczej substancji. W systemie GFG parametry jakości będą reprezentowane przez *b*-atrybuty obiektu pasywnego.

Pierwsze zadanie systemu polega na utworzeniu ciągu pierwotnego dla opisanej sytuacji problemowej. Wykonując poszczególne etapy zadania, GFG będzie realizować procesy wywołane poleceniami użytkownika. Realizując polecenia GFG drukuje ciąg złożony z komunikatów poprzedzonych znakiem @ i poleceń poprzedzonych znakiem $. Od użytkownika system żąda podania danych, które wykorzystuje jako treść kolejnych atrybutów obiektu FG formułowanego w bieżącym procesie GFG. System GFG pyta też użytkownika o zgodę na kontynuację powtarzalnych fragmentów procesu: opcja [Y][*] oznacza zgodę „Y" lub brak zgody „*" – dowolny znak. Rozmieszczenie obiektów pokazano na rysunku.

Realizacja zadania

System GFG realizuje postawione zadania w czterech następujących po sobie etapach.

Etap 1. Zgromadzenie danych i utworzenie schematów obszaru obserwacji. Informacja z rysunku będzie wprowadzona do bazy wiedzy systemu jako rezultat polecenia (GRO-OBSZAR-OBSERWACJI). Przebieg procesu jest następujący:

```
>>(GRO-OBSZAR-OBSERWACJI)
  @ Schemat obszaru:
    (<symbol-obszaru> (OPIS (VA <tekst>))
                      (KLASA (VA <wzorzec obszaru>))
                      (ELEMENTY (VA <podobszar>[<podobszar>])))
  @ WPROWADZENIE ATRYBUTOW OBSZARU R:
  $ Zawartosc OPIS.inf <- OBSZAR-RZECZNY-WISLA1
  $ Zawartosc KLASA.inf <- WR1
  @ Zawartosc ELEMENTY.inf, t.j. symbole podobszarow:
  $ Symbol podobszaru <- R1
  $ Nastepny podobszar ? [Y][*] <- Y
  $ Symbol podobszaru <- R2
  ...itd. z pozostałymi podobszarami, po czym:
  @ SCHEMAT OBSZARU R:
  (R (OPIS (VA OBSZAR-RZECZNY-WISLA1))(KLASA (VA WR1))
     (ELEMENTY (VA R1 R2 R3 R4)))
  @ SCHEMAT PODOBSZARU:
  (<symbol podobszaru> (OPIS (VA <tekst>))
                       (KLASA (VA <wzorzec podobszaru>))
                       (ELEMENTY (VA <obiekt>[<obiekt>])))
  @ PODOBSZAR R1:
  $ Zawartosc OPIS.inf <- GORNA-WISLA1
  $ Zawartosc KLASA.inf <- WR2
  @ Zawartosc ELEMENTY.inf, t.j. symbole obiektow:
  $ Symbol obiektu <- o1
  $ Nastepny obiekt ? [Y][*] <- Y
  $ Symbol obiektu <- o2
  $ Nastepny obiekt ? <- N
  @ SCHEMAT PODOBSZARU R1:
  (R1 (OPIS (VA GORNA-WISLA1)) (KLASA (VA WR2))(ELEMENTY (VA
      O1 O2)))
```

... itd. z pozostałymi podobszarami. Zawarte w pamięci schematy można wywołać poleceniem (RS ⟨sch⟩).

Etap 2. Założenie rejestru B-atrybutów manifestujących zdarzenie (PO-BAT-N) i rejestru B-atrybutów obiektów aktywnych, które według hipotezy HB mogą mieć wpływ na zdarzenie (AO-BAT-N). Są to następujące dwa procesy dialogowe:

```
>>(GRO-B-ATRYBUTY-OP)
  @ ZALOZENIE REJESTRU B-ATRYBUTOW ZDARZENIA
  $ Nazwa B-atrybutu <- TKANKA-ORGANICZNA
  $ Nastepny B-atrybut ? [Y][*] <- Y
  $ Nazwa B-atrybutu <- ZWIAZKI-AZOTOWE
  $ Nastepny B-atrybut ? [Y][*] <- n
  @ ZAWARTOSC REJESTRU PO-BAT-N:
  (BAT1 TKANKA-ORGANICZNA)
  (BAT2 PESTYCYDY)
  (BAT3 ZWIAZKI-AZOTOWE)
>>(GRO-B-ATRYBUTY-OA)
  @ ZALOZENIE REJESTRU B-ATRYBUTOW OBIEKTOW AKTYWNYCH
  @ Kontekst: B-atrybuty OP:
  (BAT1 TKANKA-ORGANICZNA)
  (BAT2 PESTYCYDY)
  (BAT3 ZWIAZKI-AZOTOWE)
  $ Nazwa B-atrybutu <- AKTYWNOSC-PRODUKCYJNA
  $ Nastepny B-atrybut ? [Y][*] <- Y
  $ Nazwa B-atrybutu <- OPAD-DASZCZU
  $ Nastepny B-atrybut ? [Y][*] <- N
  @ ZAWARTOSC REJESTRU AO-BAT-N:
  (BAT1 AKTYWNOSC-PRODUKCYJNA)
  (BAT2 OPAD-DESZCZU)
```

Etap 3. Zainicjowanie schematów wszystkich tych obiektów, których symbole zostały wprowadzone do schematu obszaru w etapie 1. Proces tego etapu jest następujący:

```
>>(INI-OBIEKTY)
  @ WSTEPNY SCHEMAT OBIEKTU
  (<symbol obiektu> (OPIS (VA <inf>))
                    (ROLA (VA A : P))
                    (KLASA (VA <wzorzec obiektu>)))
  @ ATRYBUTY OBIEKTU O1:
  $ Dane atrybutu OPIS <- WARSZTAT1
  $ Dane atrybutu KLASA <- WO1
  $ ROLA obiektu O1 [A : obiekt aktywny][P : obiekt pasywny] <- A
  (O1 (OPIS (VA WARSZTAT1))(KLASA (VA WO1))(ROLA (VA A)))
  @ ATRYBUTY OBIEKTU O2:
```

... podobnie jak dla 01; tak też wygląda przebieg dla pozostałych obiektów 03, 04, 05, 06, 07. Po zakończeniu sesji systemowej schematy obiektów można odtworzyć w pamięci systemu poleceniem:

```
>>(WYW-OBIEKTY)
  (O1 (OPIS (VA WARSZTAT1))(KLASA (VA WO1))(ROLA (VA A)))
  (O2 (OPIS (VA FARMA1))(KLASA (VA WO2))(ROLA (VA A)))
  (O3 (OPIS (VA FARMA2))(KLASA (VA WO2))(ROLA (VA A)))
  (O4 (OPIS (VA FARMA3))(KLASA (VA WO2))(ROLA (VA A)))
  (O5 (OPIS (VA FABRYKA-NAWOZOW))(KLASA (VA WO3))(ROLA (VA A)))
  (O6 (OPIS (VA ZBIORNIK-KONTROLNY))(KLASA (VA WO4))(ROLA (VA P)))
  (O7 (OPIS (VA WARUNKI-ATMOSFERYCZNE))(KLASA (VA WO5))(ROLA (VA
      A)))
```

Etap 4. Zadaniem systemu w tym etapie jest odnotowanie zachowań obiektów z danego obszaru obserwacji oraz utworzenie pierwotnego ciągu pakietów FG. Proces dialogowy tego etapu ma następujący przebieg:

```
>>(GRO-CIAG-PIERWOTNY)
@ GROMADZENIE CIAGU PIERWOTNEGO
$ Miesiac  <- 3
$ Dzien    <- 12
$ Godzina  <- 13
$ Minuta   <- 0
@ POTWIERDZENIE B-ATRYBUTOW W POSZCZEGOLNYCH OBIEKTACH
$ AKTYWNOSC-PRODUKCYJNA w O1 [Y][*] <- Y
$ OPAD-DESZCZU w O1 [Y][*] <- N
$ AKTYWNOSC-PRODUKCYJNA w O2 [Y][*] <- N
$ OPAD-DESZCZU w O2 [Y][*] <- N
$ AKTYWNOSC-PRODUKCYJNA w O3 [Y][*] <- N
$ OPAD-DESZCZU w O3 [Y][*] <- N
$ AKTYWNOSC-PRODUKCYJNA w O4 [Y][*] <- N
$ OPAD-DESZCZU w O4 [Y][*] <- N
$ AKTYWNOSC-PRODUKCYJNA w O5 [Y][*] <- Y
$ OPAD-DESZCZU w O5 [Y][*] <- N
$ TKANKA-ORGANICZNA w O6 [Y][*] <- N
$ PESTYCYDY w O6 [Y][*] <- N
$ ZWIAZKI-AZOTOWE w O6 [Y][*] <- Y
$ AKTYWNOSC-PRODUKCYJNA w O7 [Y][*] <- N
$ OPAD-DESZCZU w O7 [Y][*] <- N
$ Nastepny pakiet ? [Y][*] <- Y
.............................
```

Przebieg procesu dla kolejnych pakietów jest podobny. Zakończenie procesu następuje po wyrażeniu:

$ Następny pakiet ? [Y] [*] ⟨— N

A oto przykład zapisu ciągu pierwotnego dla naszej sytuacji problemowej:

```
>>(FLPRIN 'FI-SEQ)
((3 12 13 0)(O6 ZWIAZKI-AZOTOWE)(O1 AKTYWNOSC-PRODUKCYJNA)(O2)(O3)
 (O4)(O5 AKTYWNOSC-PRODUKCYJNA)(O7))
((3 13 13 0)(O6 TKANKA-ORGANICZNA)(O1)(O2 AKTYWNOSC-PRODUKCYJNA)
 (O3 AKTYWNOSC-PRODUKCYJNA)(O4 AKTYWNOSC-PRODUKCYJNA)(O5)(O7))
((3 14 13 0)(O6 PESTYCYDY)(O1)(O2 AKTYWNOSC-PRODUKCYJNA)
 (O3 AKTYWNOSC-PRODUKCYJNA)(O4 AKTYWNOSC-PRODUKCYJNA)
 (O5 AKTYWNOSC-PRODUKCYJNA)(O7))
((3 15 13 0)(O6 TKANKA-ORGANICZNA PESTYCYDY)
 (O1 AKTYWNOSC-PRODUKCYJNA)(O2 AKTYWNOSC-PRODUKCYJNA)(O3)(O4)
 (O5 AKTYWNOSC-PRODUKCYJNA)(O7 OPAD-DESZCZU))
((3 16 13 0)(O6 ZWIAZKI-AZOTOWE)(O1)(O2)(O3)(O4)
 (O5 AKTYWNOSC-PRODUKCYJNA)(O7 OPAD-DESZCZU))
((3 17 13 0)(O6 ZWIAZKI-AZOTOWE)(O1 AKTYWNOSC-PRODUKCYJNA)(O2)(O3)
 (O4)(O5 AKTYWNOSC-PRODUKCYJNA)(O7 OPAD-DESZCZU))
```

Przedstawiony wyżej ciąg pierwotny zawiera pakiety danych FG zebranych w okresie od 12 marca do 17 marca. Dane te, zgodnie z definicją, dotyczą zachowania wszystkich obiektów odnotowanych w obszarze obserwacji R. Zapis ciągu jest więc fotografią stanu sytuacji problemowej z danego obszaru i może pełnić rolę dokumentacji. Wnikliwe rozpatrzenie zapisu przez użytkownika systemu pozwala wyciągnąć wnioski dotyczące zależności przyczynowych w danej sytuacji. Jednak przy długich ciągach jest to zadanie żmudne i podatne na błędy, stąd też słuszne wydaje się powierzenie takiego zadania systemowi.

dokończenie na s. 26

Ogłoszenie w INFORMATYCE
dotrze przede wszystkim do profesjonalistów

JUNISOFTEX Sp. z o.o.

44-100 GLIWICE ul. Konstytucji 11,
tel.-faks 31-75-10, 31-90-81 do 88 w. 250, 272, 282, teleks 036233

JUNISOFTEX – to firma z tradycjami i najdłuższymi doświadczeniami w eksploatacji wielodostępnych systemów komputerowych Novell w kraju.

JUNISOFTEX – to najstarszy w kraju wykonawca własnych wielodostępnych systemów pracujących w sieci NetWare firmy Novell.

JUNISOFTEX – to autoryzowany reseller amerykańskiej firmy Novell.

JUNISOFTEX – to dostawca zintegrowanych systemów komputerowych działających w kilkudziesięciu firmach obejmujących w każdym przedsiębiorstwie, niezależnie od formy własności i dziedziny gospodarki: finanse, majątek obrotowy, majątek trwały i nietrwały, sprzedaż, techniczne przygotowanie produkcji oraz kadry-płace.

JUNISOFTEX – to dostawca i wykonawca autoryzowanych sieci lokalnych firmy Novell.

JUNISOFTEX – to dostawca sprawdzonego sprzętu komputerowego renomowanych firm ALR, IBM, TEAM, DIGILAB, EPSON, MANNESMANN TALLY.

JUNISOFTEX – to wyłączny i autoryzowany dystrybutor komputerów i terminali firmy DIGILAB na Śląsku.

JUNISOFTEX – to dostawca i wykonawca okablowania lokalnych sieci komputerowych ARCNET, ETHERNET na kablach zwykłych i światłowodowych oraz na dwużyłowych kablach telefonicznych w przypadku stosowania oprogramowania NetWare ACCESS SERVER firmy Novell.

JUNISOFTEX – to nauczyciel, który chętnie podzieli się swoją wiedzą na organizowanych kursach w swojej szkole informatycznej – w tym nauczy Cię: podstaw informatyki, obsługi sprzętu komputerowego, eksploatacji własnych systemów informatycznych, edytorów tekstu: CHiWRITER, WordPerfect, arkuszy kalkulacyjnych QPRO v. 3.0, LOTUS 1-2-3 itp.

JUNISOFTEX – to doradca w zakresie komputerowych metod organizacji i eksploatacji systemów komputerowych.

JUNISOFTEX – to strażnik postępu i nowoczesności w Twojej firmie.

JUNISOFTEX – to gwarancja niezawodności, rzetelności i terminowości.

JUNISOFTEX – to Twój doradca i partner, któremu możesz zaufać, który Cię nigdy nie zawiedzie.

JUNISOFTEX – to partner, który Cię wysłucha i zawsze pomoże podjąć dobrą decyzję.

Jeżeli chcesz

- ☆ zastosować najnowocześniejsze systemy komputerowe oraz lokalne sieci komputerowe,
- ☆ zreorganizować, usprawnić i szybko skomputeryzować swoje przedsiębiorstwo,
- ☆ lepiej wykorzystać już istniejące w Twoim przedsiębiorstwie komputery,
- ☆ wymienić swoje niefortunnie zakupione oprogramowanie,
- ☆ pozbyć się problemów eksploatacyjnych, upadających systemów, nie działających komputerów,
- ☆ przeszkolić załogę swojego przedsiębiorstwa,
- ☆ dobrze zainwestować swoje pieniądze i podnieść rangę swojego przedsiębiorstwa,
- ☆ w przyszłości mieć dostęp do krajowej i światowej sieci komputerowej,

TO ZGŁOŚ SIĘ DO NAS!

0/17/91

DARIUSZ CABAN
JACEK JARNICKI
Instytut Cybernetyki Technicznej
Politechnika Wrocławska

Rekonfigurowalny filtr obrazów cyfrowych

Zagadnienie filtrowania obrazów cyfrowych, czyli takich, w których informacja jest zapisana w postaci tablicy liczb, zostało dobrze opracowane teoretycznie [2, 3]. Znane są liczne algorytmy realizujące filtry o bardzo różnych właściwościach, np. „polepszające" obraz przez eliminowanie zakłóceń wielkiej częstotliwości lub dokonujących kompresji dynamiki obrazu, pozwalającej – po stłumieniu częstotliwości w pobliżu składowej stałej – na zapisanie obrazu odfiltrowanego w mniejszej jednostce pamięci, bez straty kontrastu jego szczegółów.

Dziedziny zastosowań filtracji obrazów cyfrowych są bardzo różne – od radiologii cyfrowej, w której wymaga się ogólnego polepszenia jakości obrazu, czytelności szczegółów i poprawy kontrastu, do wizyjnych systemów robotyki, gdzie podstawowym zadaniem filtru jest eliminowanie zakłóceń pochodzących od oświetlenia sceny i wnoszonych przez tor wizyjny.

Sformalizowanie opisanych w literaturze algorytmów filtrowania umożliwia dość prostą realizację programową. Niestety, w wielu przypadkach programy działają zbyt długo. Przykładowo, dla zastosowań w systemach robotyki, od których jest wymagana reakcja w czasie rzeczywistym, a sama filtracja jest jedynie niewielkim fragmentem procesu przetwarzania obrazu, filtrowanie programowe jest nie do przyjęcia.

Należy zaznaczyć również, że pożądany efekt filtracji można uzyskać po zastosowaniu właściwie wybranego algorytmu. Wymaga to pewnego doświadczenia i znajomości właściwości różnorodnych filtrów. Przy konstruowaniu filtrów należy więc dążyć do spełnienia dwóch wymagań: po pierwsze szybkości umożliwiającej uzyskanie obrazu odfiltowanego w czasie rzeczywistym, a po drugie elastyczności rozumianej jako łatwość modyfikacji algorytmu filtracji.

Celem niniejszego artykułu jest przedstawienie rozwiązania sprzętowego spełniającego w dużej mierze oba powyższe wymagania. Opisany rekonfigurowalny filtr obrazów realizowany jest za pomocą układów programowalnych matryc logicznych LCA, co ponadto stanowi okazję do zaprezentowania możliwości tych jeszcze niezbyt dobrze znanych w kraju elementów. W rozważaniach ograniczono się do dwóch wybranych typów algorytmów filtracji. Przeanalizowano realizację algorytmu filtru liniowego o zmiennych parametrach i tzw. filtru środkowego (medianowego), będącego przykładem filtru nieliniowego.

Algorytmy filtracji obrazów

Niech x_{ij} oraz y_{ij} będą liczbami dodatnimi z pewnego zakresu i oznaczają stopień jaskrawości punktu o współrzędnych (i, j) odpowiednio obrazu wejściowego i obrazu odfiltrowanego.

Zadanie filtracji polega na wyznaczeniu dla każdego punktu o współrzędnych (i, j) wartości y_{ij} wyliczanej jako wynik operacji wykonywanej na liczbie x_{ij} i liczbach opisujących sąsiednie punkty obrazu wejściowego. W ogólności obszarem sąsiedztwa może być nawet cały obraz.

Przykładem takiej operacji jest filtr liniowy opisany równaniem:

$$y_{i,j} = \sum_{k=-N_1}^{N_1} \sum_{l=-N_2}^{N_2} h_{kl}\, x_{i+k,j+l}$$

Funkcja h_{kl} jest tzw. jądrem filtru, a liczby N_1, N_2 definiują obszar sąsiedztwa wokół punktu o współrzędnych (i,j), zwany oknem. Jest to szczególny przypadek filtru liniowego i nosi nazwę niezmienniczego względem przesunięcia. Funkcja h_{kl} zależy jedynie od rozmiarów analizowanego okna, a nie od jego położenia na obrazie.

Programowa realizacja takiego filtru jest dość prosta, jednak czas obliczeń przy dużych rozmiarach obrazu i okna może okazać się znaczny.

Bardziej skomplikowane obliczeniowo jest zadanie filtracji nieliniowej, w którym definicja operacji wyznaczającej y_{ij} zależy od wartości punktów obrazu wejściowego. Typowym zadaniem tej klasy jest filtr środkowy, w którym operacja obliczania y_{ij} jest zadana następująco:
1) uporządkować elementy okna, czyli liczby $x_{i+k,j+l}$ w ciąg monotoniczny;
2) jako wartość punktu obrazu odfiltrowanego y_{ij} przyjąć środkowy element wyznaczonego ciągu.
Mimo prostoty opisu, realizacja programowa filtru środkowego jest nadzwyczaj czasochłonna, wymaga bowiem dla każdego punktu obrazu wykonania uciążliwego sortowania elementów okna. Znane są algorytmy w pewnym stopniu przyspieszające obliczenia. Wykorzystuje się w nich uporządkowanie uzyskane dla punktu poprzedniego jako częściowe dane dla punktu następnego.

Programowalne matryce logiczne

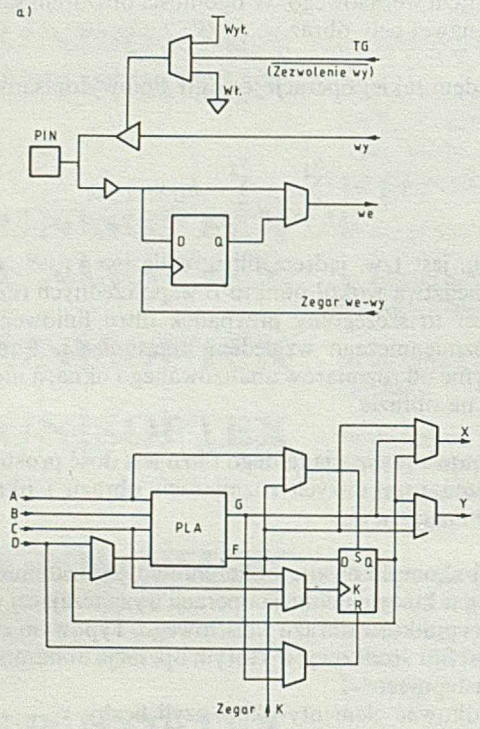
Programowalne matryce logiczne są to dość nowe układy o wielkim stopniu scalenia. Jako pierwsze w tej grupie pojawiły się układy 2064 i 2018, opracowane wspólnie przez Xilinx Inc. oraz Monolythic Memories [4, 5]. Obecnie są również dostępne układy innych producentów (AMD, Actel, Texas Instruments). W miarę pełny przegląd tych układów można znaleźć w [1].

Programowalne matryce logiczne są to tablice jednakowych, konfigurowalnych bloków funkcjonalnych (nazywanych komórkami).
W układach tych programuje się:
- funkcje realizowane przez poszczególne komórki,
- połączenia między komórkami.

Układy można programować wielokrotnie, bez konieczności

kasowania (jak pamięć RAM). Ma to szczególne znaczenie w omawianym zastosowaniu, umożliwia bowiem zmiany algorytmu filtracji obrazu bez modyfikacji sprzętowych.

Zależnie od typu układu matrycowego, komórki mogą mieć różne konfiguracje. W dalszych rozważaniach jest opisywana rodzina układów CLB 2064/2018. W ich skład wchodzą matryce 8 × 8 (układ 2064) lub 10 × 10 (układ 2018) komórek podstawowych, oznaczanych jako CLB, oraz odpowiednio 58 lub 74 komórki wejściowe-wyjściowe oznaczane jako IOB. Na rysunku 1 przedstawiono schematy tych komórek. Jest w nich programowana funkcja PLA oraz multipleksery, oznaczone na rysunku trapezami. Matryca PLA może być używana do realizacji jednej funkcji 4-wejściowej lub dwóch funkcji 3-wejściowych.



Rys.. 1. Schematy komórek układu LCA: a) komórka IOB, b) – komórka CLB

Komórki IOB pozwalają dowolnie definiować funkcję większości wyprowadzeń matryc logicznych. Mogą to być wejścia lub wyjścia proste, trójstanowe lub buforowane, a także wyprowadzenia dwukierunkowe, sterowane sygnałem wewnętrznym.

Połączenia między komórkami są realizowane przez blokowanie lub uaktywnianie ścieżek znajdujących się w strukturze układu matrycowego. Liczba tych ścieżek jest ograniczona, tak że projekt topologii połączeń stanowi nietrywialny problem przy wykorzystaniu układów LCA. Występują podane niżej trzy typy połączeń.

Połączenia bezpośrednie. Mogą one łączyć sąsiednie komórki w pionie i w poziomie. Jest to najprostszy typ połączeń, mający jedynie ograniczone zastosowanie.

Połączenia przez szyny. Programuje się połączenie jednego wyjścia komórki na sąsiadującą z nią szynę oraz połączenia dowolnej liczby wejść do komórek z tą szyną. Z każdym wierszem komórek sąsiadują trzy szyny poziome, a z każdą kolumną – cztery pionowe. Dodatkowo jest również dostępna szyna zegara taktującego połączona w obrębie całego układu. Tego rodzaju połączenia są stosowane do łączenia odległych komórek oraz doprowadzania sygnałów do wielu komórek jednocześnie.

Połączenia przez bloki komutacyjne. Na skrzyżowaniu każdego wiersza i kolumny matrycy komórek jest umieszczony (w strukturze układu LCA) blok komutacyjny. Programować można połączenia bloków komutacyjnych z sąsiadującymi komórkami, a także połączenia między blokami komutacyjnymi. W ten sposób można zestawiać połączenia dowolnej długości z segmentów między sąsiednimi blokami komutacyjnymi. Oczywiście liczba segmentów między blokami komutacyjnymi jest ograniczona, co utrudnia zestawianie tego rodzaju połączeń.

Istotnym utrudnieniem przy projektowaniu połączeń jest fakt, że wnoszą one opóźnienia propagacji sygnałów. Wielkość tych opóźnień zależy od typu połączeń, ich obciążenia, jak również od liczby segmentów, z których są złożone. W przypadku połączeń innych niż bezpośrednie, opóźnienia są porównywalne z szybkością działania komórek.

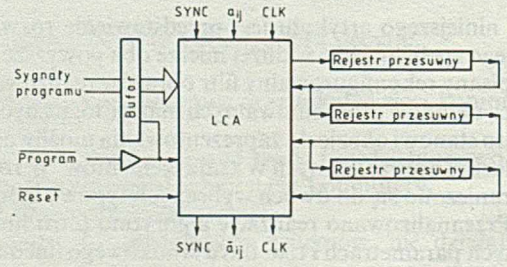
Programowanie układów LCA odbywa się przez wpisywanie odpowiedniej informacji konfiguracyjnej. Przygotowanie tej informacji stanowi poważny problem. Jego rozwiązanie umożliwia system wspomagania projektowania o nazwie XACT [4], produkcji firmy Xilinx Inc. Projektowanie odbywa się interakcyjnie, podobnie jak w systemach wspomagających projektowanie płytek drukowanych. W wyniku powstaje zbiór informacji konfiguracyjnej w formacie JEDEC lub szesnastkowym. Informacja konfiguracyjna (ok 4 KB) może być wczytana do układu LCA na wiele różnych sposobów. Po włączeniu zasilania układ LCA może automatycznie odczytać informację konfiguracyjną z przyłączonej do niego pamięci EPROM. W innym trybie informacja jest wprowadzana szeregowo z zewnętrznym taktowaniem. Umożliwia to bezpośrednie wpisywanie informacji z komputera, dla którego układ LCA może być normalnym portem.

Struktura filtru sygnału wizyjnego

Wykorzystując właściwości wielokrotnego programowania układów LCA można zrealizować rekonfigurowalny filtr sygnału wizyjnego.

Architektura filtru

Ogólnie, wszystkie tego rodzaju filtry wymagają pamiętania informacji w oknie $(2N_1 + 1) \times (2N_2 + 1)$ punktów (najczęściej 3 × 3). Zapewnia to układ przedstawiony na rys. 2. Funkcja filtrująca jest realizowana w tym oknie, całkowicie w obrębie układu LCA. Dlatego do zmiany funkcji jest konieczne wpisanie nowej informacji konfigurującej układ LCA.



Rys. 2. Schemat rekonfigurowalnego filtru obrazu

Informacja konfigurująca jest wprowadzana z systemu mikrokomputerowego poprzez układy buforujące. Od strony mikrokomputera są one widoczne w przestrzeni portów. Bufory są normalnie w stanie wysokiej impedancji od strony układu LCA. Pozwala to wykorzystać te same wyprowadzenia układu LCA do programowania i w normalnej pracy filtru. Uaktywnienie buforów powoduje jednocześnie wysterowanie sygnału wprowadzającego układ LCA w stan konfigurowania.

Filtr przetwarza sygnał wizyjny w czasie rzeczywistym, generując na wyjściu sygnał odfiltrowany. Opóźnienie sygnału w stosunku do wejścia wynosi N_1 linii $+N_2$ punktów. Opóźnienie między dostępnością punktów w oknie wynosi $2N_1 + 1$ linii. Realizowane musi być na układach zewnętrznych – rejestrach przesuwnych o pojemności $2N_1 + 1$ linii.

W obrębie układu LCA wykorzystuje się $M \star (2N_1 + 4)$ wyprowadzeń na wejście i wyjście wideo oraz na interfejs z rejestrami przesuwnymi (M jest liczbą bitów stanu jaskrawości punktu). Dla $M = 4$ oraz okna 3×3 wynosi to 24 wyprowadzenia. Dalsze cztery wyprowadzenia zajmują sygnały taktujące. Wykorzystuje się $(2N_1 + 1) \star 2N_2 \star M$ przerzutników częściowo w blokach CLB, częściowo w IOB. W sumie dla okna 3×3 wykorzystanych jest 28 bloków ION i dodatkowe 24 przerzutniki (bloki CLB lub IOB).

Oprócz tych bloków konieczne są komórki, na których jest zrealizowany algorytm filtrujący. Aby zapewnić elastyczność jego konfigurowania, trzeba zastosować układ LCA typu 2018 (100 bloków CLB).

Realizacja filtru liniowego

Aby uzyskać określone właściwości filtru, układ LCA musi być odpowiednio skonfigurowany. Przygotowanie danych konfiguracyjnych polega na:

- przygotowaniu schematu logicznego filtru (w postaci sieci bramek i rejestrów),
- przekształceniu tej sieci na listę bloków CLB oraz połączeń między nimi (wykorzystuje się do tego gotowe biblioteki makrokomórek),
- zaprojektowaniu rozmieszczenia bloków oraz połączeń w obrębie układu LCA (za pomocą pakietu XACT [4]).

Rozważmy przykładową realizację filtru liniowego. Aby oszacować konieczną liczbę bloków CLB zakłada się, że poszczególne punkty obrazu są kodowane słowami 3-bitowymi (8 poziomów jaskrawości). Ponadto zakłada się, że filtr jest realizowany z wykorzystaniem arytmetyki całkowitoliczbowej, tzn. współczynniki jądra filtru są w postaci $h_{kl} = a_{kl}/2^M$. Przy przyjęciu rozmiarów okna 3×3, równanie opisujące filtr ma postać:

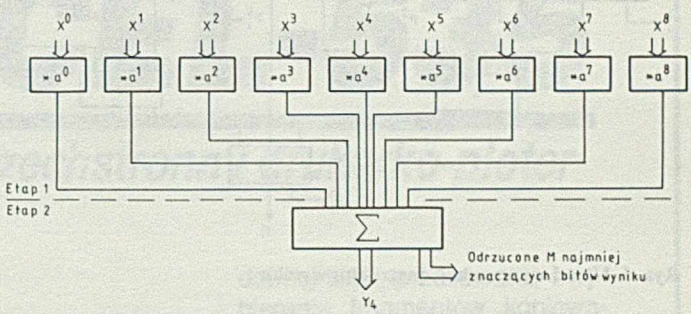
$$y_{i,j} = \sum_{k=-1}^{1} \sum_{l=-1}^{1} a_{kl} x_{i+k,j+l}$$

przy czym a_{kl} są liczbami całkowitymi 4-bitowymi, x_{ij}, y_{ij} liczbami 3-bitowymi, natomiast 2^M jest wielkością normalizującą wynik filtracji (np. $M = 4$).

Układ logiczny realizujący powyższe wyrażenie arytmetyczne jest dość rozbudowany. Można go znacznie uprościć, przyjmując konkretne wartości liczb a_{kl}. Wymagałoby to jednak odrębnego projektowania filtrów dla każdej szczególnej postaci jądra. Dlatego zdecydowano się na realizację układu w postaci ogólnej, bez narzucania ograniczeń na wartości współczynników a_{kl}.

Aby uzyskać łatwą możliwość zmiany tych współczynników, filtr jest projektowany tak, aby współczynniki wchodziły do równań opisujących funkcje poszczególnych bloków CLB. Nie mogą one wpływać na układ połączeń między blokami.

Filtr jest realizowany w dwóch etapach. W pierwszym etapie wykonuje się mnożenie słów 3-bitowych (jaskrawości punktów obrazu) przez stałe współczynniki funkcji jądra. W wyniku powstają 7-bitowe słowa. Są one w drugim etapie sumowane, a M najmniej znaczących bitów wyniku odrzuca się. Ogólną strukturę filtru pokazano na rys. 3.



Rys. 3. Ogólna struktura filtru

Mnożenie przez stałą jest operacją dość prostą. Wykorzystuje się w tym przypadku następujące wzory:

$$S_i^k = \sum_{j=\max\{i-3,0\}}^{\min\{i,4\}} a_j^k x_{i-j}^k + (S_{i-1}^k/2) \bmod 2 + (S_{i-2}^k/4) \bmod 2$$

$$\text{dla } i = 0,1, ..., 6,$$

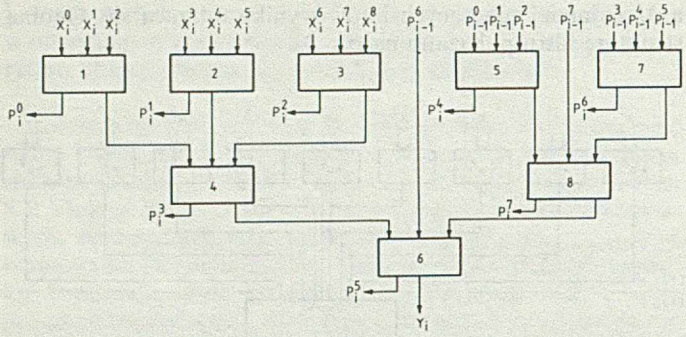
przy czym:
x_j^k – j-ty bit słowa kodowego,
a_j^k – j-ty właściwego współczynnika jądra (mnożnika a_{kl}),
S_i^k – suma bitów,
$S_i^k/2$, $S_i^k/4$ – bardziej znaczące bity sumy (operacja dzielenia całkowitoliczbowego),
k – numer punktu okna ($k = 0..8$)..
Aby uzyskać i-ty bit obliczanego słowa 7-bitowego, należy wziąć najmniej znaczący bit sumy, tzn. $S_i^k = (S_i^k) \bmod 2$.
Sumy S_i^k otrzymuje się przez dodanie od 1 do 6 wartości binarnych. Są zatem zawsze mniejsze niż 6, a więc przeniesienie jest najwyżej 2-bitowe. Łatwo zauważyć, że $S_0^k = a_0^k x_0^k$ nie wymaga do realizacji żadnego bloku CLB (mnożenie przez współczynnik a_0^k można przenieść do funkcji realizowanych w blokach sumatora w drugim etapie realizacji filtru)..

S_1^k i S_2^k są funkcjami logicznymi odpowiednio dwóch (x_0^k, x_1^k) i trzech (x_0^k, x_1^k, x_2^k) zmiennych, a zatem do ich realizacji wystarcza jeden blok CLB. Pozostałe S_3^k, ..., S_6^k są funkcjami zmiennych x_0^k, ..., x_3^k i każda z nich wymaga osobnego bloku CLB. W oknie 3×3 mnoży się na raz 9 słów kodowych, a zatem pierwszy etap algorytmu jest realizowany przez 45 bloków CLB. Bloki te są ułożone regularnie, co oczywiście ułatwia projektowanie połączeń.

W drugim etapie sumuje się 9 słów 7-bitowych. Zakłada się przy tym, że filtr jest dobrany właściwie i wynik nigdy nie przekracza zakresu liczb 7-bitowych. Sumator jest zrealizowany z użyciem elementarnych sumatorów 3-bitowych, generujących bit wyniku oraz przeniesienie 1-bitowe. Każdy sumator elementarny jest realizowany przez jeden blok CLB. Na rys. 4 przedstawiono układ wyliczający 1 bit wyniku. Układ sumuje odpowiednie bity 9 słów wejściowych wyznaczonych w pierwszym etapie (oznaczone $s_i^0..s_i^0$) oraz przeniesienia z sumowania bitów o jeden mniej znaczących (oznaczane $p_{i-1}^0...p_{i-1}^8$). Przeniesienia z sumowania kolejnych bitów ($p_i^0..p_i^3$) są przekazywane do układu bardziej znaczącego bitu sumatora.

Jak widać, do realizacji większości bitów sumy potrzeba po 8 bloków CLB. Wyjątkiem są bity:

- najmniej znaczący (bit 0), w którego przypadku nie ma przeniesień z pozycji mniej znaczących; wystarczają zatem bloki oznaczone 1–4 (rys. 4);
- bit 1, w którego przypadku występują jedynie przeniesienia p_0^0; p_0^3, a zatem wystarczają bloki 1–6 (p_0^3 i wyjście z bloku 5 są dołączone bezpośrednio do bloku 6);



Rys. 4. Układ wyliczania jednego bitu wyniku

● bit 2, gdzie nie występują przeniesienia p_1^6 i p_1^7, a zatem zbędny jest blok 8.

Reasumując, sumator 7-bitowy wymaga 49 bloków CLB. Ponieważ M najmniej znaczących bitów wyniku nie jest wyprowadzanych na zewnątrz więc M z tych bloków może być równocześnie wykorzystywane jako przerzutniki.

Realizacja 3-bitowego filtru liniowego wymaga w sumie 94 bloków CLB (bez bloków CLB i IOB). W żadnym z bloków nie wykorzystuje się przerzutnika. Brak jest jednak wolnych wejść, które umożliwiłyby użycie większości z tych przerzutników do pamiętania jaskrawności punktów okna obrazu. Wykorzystując część bloków IOB jako przerzutniki, filtr 3-bitowy można zrealizować w pojedynczym układzie 2018. Filtracja sygnału 4-bitowego wymaga już jednak dwóch układów 2018 (lub układu rodziny 3000).

Przykładowa realizacja filtru środkowego

Aby ocenić złożoność realizacji filtrów nieliniowych, w omawianej konfiguracji zaimplementowano algorytm filtru środkowego (wyznaczania mediany). Wykorzystano następujący algorytm:
– najbardziej znaczący bit mediany jest wyznaczany bezpośrednio jako większość spośród najbardziej znaczących bitów jaskrawości punktów w obrębie okna,
– jeżeli określony punkt ma jaskrawość mniejszą niż mediana, co wynika z faktu, że na $(i-1)$ najbardziej znaczących bitach jest równy medianie, a w i-tym bicie ma wartość 0, gdy tymczasem mediana ma wartość 1, to wszystkie bity mniej znaczące od i są zerowane; jeżeli ma jaskrawość większą niż mediana na i-tym bicie, to wszystkie bity mniej znaczące od i są ustawiane na 1;
– kolejne bity mediany są wyznaczane podobnie, jak bit najbardziej znaczący, przy czym bierze się pod uwagę wartości słów zmodyfikowane według powyższych zasad.

Układ logiczny modyfikujący i-ty bit jaskrawości punktów ma postać:

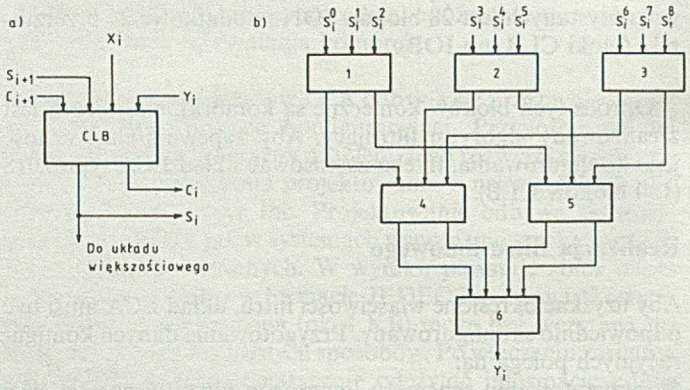
$$s_i^k = x_i^k \star \bar{c}_{i+1} + s_{i+1}^k \star c_{i+1},$$
$$c_i = c_{i+1} + y_i \oplus x_i^k,$$

przy czym:
x_i^k – i-ty bit jaskrawości punktu w oknie,
c_i – zmienna pomocnicza, określana dla każdego bitu na podstawie bitów bardziej znaczących, a dla najbardziej znaczącego równa 0,
y_i – wartość i-tego bitu mediany (jaskrawości w środku okna po filtracji),
operacja $\oplus$ oznacza różnicę symetryczną, a – negację logiczną.

Mediana 1-bitowa (układ większościowy) jest zrealizowana (dla okna 3 × 3) jako:

$$y_1 = \begin{cases} 1, & \text{gdy } \sum_{k=0}^{8} s_i^k > 4, \\ 0 & \text{w przeciwnym razie} \end{cases}$$

Realizację filtru środkowego przedstawiono na rys. 5. Każdy bit jaskrawości punktów okna jest modyfikowany przez jeden blok CLB, realizujący opisaną wyżej funkcję. Jeden bit mediany otrzymuje się w układzie z rys. 5b, przy czym bloki 1–5 realizują funkcję sumatorów podobną jak w filtrze liniowym. Blok 6 wylicza ostateczną sumę i porównuje z progiem.



Rys. 5. a) realizacja filtru środkowego, b) układ wyliczania jednego bitu mediany (jednego bitu wartości jaskrawości środka okna na wyjściu filtru środkowego)

Jak widać, do realizacji filtru w oknie 3 × 3 o 16 poziomach szarości (4 bitowe słowa kodowe) potrzeba 60 bloków CLB (36 do modyfikacji każdego bitu słów wejściowych oraz do wyznaczenia median 1-bitowych).

LITERATURA

[1] Bolton M.: Introduction and PGA bibliography. Microprocessors and Microsystems. Vol. 13, No. 5, June 1989
[2] Pavlidis T.: Grafika i przetwarzanie obrazów. WNT, Warszawa 1987
[3] Pratt W.: Digital Image Processing. Wiley, New York 1979
[4] The Programmable Gate Array Data Book. Xilinx Inc. 1989
[5] Progammable Gate Arrays 1989/1990 Data Book. Advanced Micro Devices AN-182.

W związku z wprowadzeniem powszechnego podatku dochodowego i koniecznością informowania właściwych Urzędów Skarbowych o wszystkich wypłatach zwracamy się do Autorów wysyłających teksty do opublikowania o podawanie następujących danych:

- **nazwisko, imiona (pierwsze i drugie),**
- **imiona: ojca i matki,**
- **miejsce i data urodzenia,**
- **numer identyfikacyjny PESEL (wpisany przez Biuro Meldunkowe do dowodu osobistego – nie mylić z numerem dowodu osobistego!),**
- **dokładny adres (miejsce zameldowania),**
- **adres Urzędu Skarbowego właściwego dla miejsca zamieszkania Autora (bardzo ważne – bez tej informacji nie możemy przekazać honorarium do wypłaty!).**

Prosimy także – do naszej wiadomości – podać numer telefonu służbowego i ustalić formę przekazania honorarium (kasa Wydawnictwa, poczta, konto).

AMT ACCEL-535

– to najszybszy i najbardziej wszechstronny drukarko-ploter

Intelli-Plot TURBO

Przygotuj się do nowego, fascynującego sposobu kreślenia projektów przy pomocy Intelli-Plot Turbo produkowanego przez AMT (Advanced Matrix Technology). Wzięliśmy w pełni wyciszoną, pracującą na formacie do A2, drukarkę AMT ACCEL-535, dodaliśmy język graficzny Hewlett-Pacarda (HP-GL) oraz procesor AutoCAD ADI. Rezultatem jest najszybszy i najbardziej wszechstronny drukarko-ploter.

Intelli-Plot Turbo dzięki szeregowym i równoległym interfejsom może być łączony szybko i bez wysiłku z PC, mini, mainframe albo wybraną stacją CAD. Po prostu prześlij rysunki ze swojego niezależnego programu do Intelli-Plot Turbo, a on je wykreśli szybciej niż jakikolwiek ploter DTP. Przy tym nie potrzeba żadnych plików pośrednich, dodatkowych programów zajmujących pamięć stałą, ani dodatkowego hardware'u.

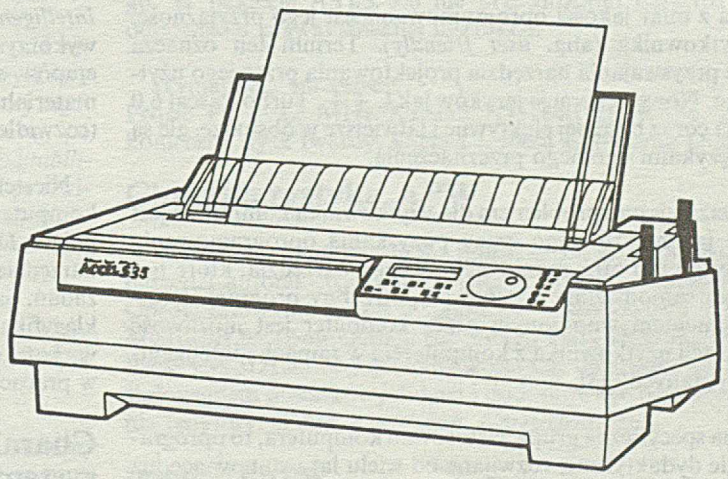
Intelli-Plot Turbo proponuje Ci rozszerzoną listę możliwości, których nie znajdziesz w ploterach DTP, zaczynając od tego, że ma różne możliwości (szybkość, jakość) kreślenia; niskojakościowe do wydruku kontrolnego, jakość średnia oraz dokładniejsze, wysokojakościowe do wyjątkowo przejrzystych i precyzyjnych wykresów.

Intelli-Plot Turbo ma także wbudowaną możliwość drukowania aż do 20 rysunków jednocześnie, w żądanej kolejności. Istnieje też możliwość zmian w Twoim projekcie bez ponownego użycia komputera, powiększanie lub zmniejszanie w nieprawdopodobnej skali, przesuwanie i wybieranie fragmentów kopiowania, skalowanie automatyczne i ustawianie w żądanym formacie. Możesz wybierać z palety 16 kolorów, 3 grubości linii dla każdego z 15 pisaków.

Intelli-Plot drukuje na różnych rodzajach papieru włącznie z pergaminem. Do ładowania papieru możesz używać wbudowanego podajnika, podajnika dodatkowego z możliwością ładowania papieru w formacie do A2 albo podawać papier z roli.

DANE TECHNICZNE:

- **emulacja HP-GL HP-7475 oraz Epson, IBM, DEC etc.**
- **MTBF 15.000 godz.**
- **format A2**
- **druk w 16 kolorach**



Advanced Matrix Technology Inc.

KUPON Inf. 6/92

Zainteresowanych dodatkowymi informacjami związanymi z naszą ofertą, prosimy o nadsyłanie wypełnionego kuponu pod adresem: **ABC Data – 00-865 Warszawa, ul. Wallców 13.**

Imię i Nazwisko .....

Zakład pracy, stanowisko .....

.....

ul. .....

Kod ..... Miasto .....

ABC DATA

Przedstawicielstwo w Polsce:
00-865 Warszawa
ul. Wallców 13
tel. 24-11-43, 24-78-35
fax 24-12-83, tlx 825098

31-066 Kraków
ul. Skawińska 11
tel. 21-98-92
fax 21-96-80, tlx 326497

358/92

Praca jest realizowana w ramach projektu badawczego nr 8-0275-91-01 finansowanego przez Komitet Badań Naukowych.

JAN PIECHA
ANDRZEJ PRZYBYLSKI
JAROSŁAW ZYGUŁA
Uniwersytet Śląski
Zakład Elektroniki i Informatyki
Katowice

APW
graficzne narzędzia programowania

Wzrost stopnia złożoności obiektów symulacji sprawia, że klasyczne języki programowania nie dorównują rosnącym potrzebom. Symulacja komputerowa jest ważną częścią wielu przedsięwzięć projektowych ponieważ daje możliwość doskonalenia projektu bez potrzeby budowy modelu fizycznego. Algorytmiczność i wielowariantowość rozwiązywania problemów sprawia, że komputer spełnia bardzo istotną rolę w procesie kształcenia, nie tylko projektantów tego procesu, ale i na każdym innym szczeblu edukacji. W artykule pokazano przykład współczesnego narzędzia symulacji – *Authorware Professional for Windows*. Ułatwia ono projektowanie materiałów wspomagających nauczanie.

Współczesne badania informatyków dotyczą w znacznej mierze obiektowych graficznych języków programowania. Nakłady na systemy komputerowego wspomagania projektowania (CAD) są automatycznie przenoszone na narzędzia i metody projektowania systemów komputerowego nauczania (CAT, CBT, CAL).

Jedną z miar jakości oprogramowaniajest jego przyjazność. dla użytkownika (ang. *user friendly*). Termin ten oznacza łatwość przyswajania narzędzia projektowania przez jego użytkownika. Nowe generacje języków jak C++, Turbo Pascal 6.0 i inne, są coraz bardziej efektywne i łatwiejsze w obsłudze, ale są ciągle językami ogólnego przeznaczenia.

Najważniejszym problemem dla użytkownika komputera jest kwestia jak najkrótszego czasu pozyskania oprogramowania użytkowego lub otrzymania efektywnego narzędzia, które pozwoli mu samodzielnie opracować potrzebny program. Istotnym elementem wnoszonym przez komputer jest możliwość konwersacji użytkownika z komputerem w ramach „inteligentnej bazy danych" [3].

Pewna specyficzna grupa zastosowań komputera, to oprogramowanie dydaktyczne, rozwijane od wielu lat i stanowiące już daleko zaawansowaną dziedzinę badawczą [6].

Systemy autorskie (ang. *authoring systems*) zawierają zestaw narzędzi programowania. Są one istotnym krokiem na drodze poszukiwań przyjaznego oprogramowania upraszczającego proces opracowania programów dydaktycznych.

Użytkownik operuje między innymi: interaktywnym edytorem graficznym, edytorem tekstów, narzędziami animacji obrazu, edytorem dźwięku, konwersacyjną bazą danych oraz importerami obrazów i dźwięków analogowych (techniką multimedialną). Użytkownik redaguje elementy opracowania w formie elementarnych obiektów oraz rejestruje je w bibliotece systemu, z możliwością wielokrotnego wykorzystania. Technika menu (ang. *menu driven system*) prowadzi projektowanta przez kolejne szczeble projektowania.

Wprowadzenie w tematykę narzędzi symulacji wymaga określenia istotnych różnic między trzema podstawowymi klasami oprogramowania dydaktycznego:

CAI (ang. *Computer Assisted Instruction*) oznacza programy prezentacyjne, w których konwersacja z komputerem nie jest sprawą istotną, ponieważ są używane przez nauczyciela jako element ilustracji zajęć;
CBT (ang. *Computer Based Teaching*) oznacza programy używane również przez nauczyciela, lecz o znacznych możliwościach rozwidleń kursu oraz z elementami konwersacji z komputerem;
CAL (ang. *Computer Aided Learning*) oznacza programy przeznaczone do samodzielnego uczenia się, wyposażone w bogaty system konwersacji i wartościowania odpowiedzi uczącego się. Zamiast CAL jest stosowana również nazwa ITS (ang. *Intelligent Tutoring System* – inteligentny system uczący), wykorzystujący tzw. bazę wiedzy. Na podstawie kolejnych etapów sprawdzania poziomu opanowania przekazywanego materiału są automatycznie wybierane dalsze fazy kształcenia (rozwidlenia tematyczne, poziom szczegółowości itp.).

Niestety większość autorów publikacji na temat zastosowań komputerów w nauczaniu nie rozróżnia wymienionych wyżej pojęć. Dlatego omówienie cech oraz analiza przydatności narzędzia programowania winna być poprzedzona określeniem zadań, jakie za ich pomocą zamierzamy wykonać. Ogólną klasyfikację systemów kształcenia wspomaganego komputerem w kategoriach programowanego nauczania przedstawiono w pracach [5, 6].

Charakterystyka przykładowych systemów autorskich

Poniżej przedstawiono charakterystykę trzech przykładowych systemów autorskich o różnych stopniach złożoności. Szczegółowo omówiono najbardziej zaawansowane rozwiązanie systemu APW (ang. *Authorware Professional for Windows* – profesionalnego systemu autorskiego dla środowiska Windows 3.0).

ORGUE/SHIVA

SHIVA jest zintegrowanym pakietem autorskim służącym do projektowania interaktywnych, multimedialnych programów

uczących. Pakiet opracował międzynarodowy zespół autorów Wspólnoty Europejskiej w ramach programu badawczego DELTA [2]. Projekt oprogramowania można zrealizować w następujących trzech trybach środowiska SHIVA:
- graficznym, ze specyfikacją sekwencji uczących oraz interakcji uczącego się,
- trybie języka ogólnego przeznaczenia (Pascal lub C),
- kombinacji wyżej wymienionych trybów.

Projekt materiałów CAL/CAI, realizowany za pomocą pakietu SHIVA przechodzi przez następujące etapy:

1. Edycji multimedialnej:
– edytor graficzno-tekstowy z importerem obiektów „bitmap",
– edytor dźwięków z digitalizacją głosu i muzyki (komendy: *cut, copy, paste, mixing* z opcją play-back),
– edytor Video/Photo z digitalizacją ruchomych ekspozycji wideo i zintegrowanej grafiki na monitorze VGA.

2. Deklaracji sekwencji materiałów dydaktycznych:
– sieci działań programu (prezentacji informacji) z wariantami prezentacji rodzajów testów według zasady *case of*,
– interakcji (22 programowane warianty): klawisze myszy, (single/double-click), wybór z menu, określone klawisze, teksty, formuły gramatyczne, wybrane kolory, pola, itp.

3. Debuger materiałów CAL/CAI.

Wymagania sprzętowe: PC 286/386, PS-2, MS DOS 3.2, karta graficzna EGA/VGA, mysz, 640 kB RAM, dysk twardy, interfejs Videologic – DVA4000 oraz karta sterownika dźwięku: Xcom-Media-Sound 3000/7000.

AUTHORWARE PROFESSIONAL FOR WINDOWS (APW)

APW jest obecnie najbardziej zaawansowanym systemem autorskim. Zawiera on następujące narzędzia projektowania:

1. Edytor sekwencji programu, zawierający paletę ikon, które można zestawiać w dowolny ciąg prezentacji i interakcji.

2. Edytor graficzno-tekstowy, służący do projektowania zasadniczych materiałów prezentacyjnych. Program użytkowy pozostaje w formie otwartej z możliwością bezpośredniej ingerencji (korekty) użytkownika zgodnie z jego potrzebami, po czym jest możliwa kontynuacja prezentacji (ang. *direct aditability*). Z uwagi na ograniczone możliwości graficzne APW przewidziano możliwość importu innych narzędzi graficznych ze środowiska systemu Windows (*bit-mapped, object oriented*). APW jest wyposażony w kompletny edytor tekstowy.

3. Narzędzia animacji, umożliwiające niezależny ruch wielu obiektów z koordynacją parametrów ruchu, rotacją i wzajemną interakcją między tymi obiektami.

4. System analizy odpowiedzi, umożliwiający m.in. analizę tekstów, reakcję na klawisze myszy i klawiatury komputera oraz klawisze symulowane graficznie (menu, limit czasu, ciąg interakcji i rozwidleń itp.).

5. Zmienne i funkcje, wzbogacające system interakcji, inteligencję bazy danych oraz możliwości obliczeniowe oprogramowania użytkowego. Możliwy jest import oprogramowania aplikacyjnego opracowanego w innych środowiskach programowych lub w językach ogólnego przeznaczenia.

Wymagania sprzętowe:

PC 286/386, MS DOS 3.2, MS-Windows 3.0, karta graficzna EGA lub VGA, mysz, 4 MB RAM, dysk twardy, interfejs Videologic, oraz karta sterownika dźwięku Soundblaster.

ASYS 3.0

Jest to system autorski opracowany w latach 1980–1990 w Zakładzie Elektroniki i Informatyki Uniwersytetu Śląskiego [6]. Jest to obiektowo zorientowany zestaw narzędzi projektowania programów demonstracyjnych CAI. Projektowanie materiałów CAI odbywa się dwupoziomowo (techniką buttom-up): poziom niższy – opracowanie ekranów i obiektów bibliotecznych, poziom wyższy – etap konsolidacji elementów programu.

Niski poziom programowania

Jest to etap edycji graficznej ekranów stanowiących elementarną jednostkę programu. Niski poziom programowania jest reprezentowany przez program *EDYTOR*, zawierający komendy obsługi bazy danych (biblioteki obiektów i ekranów) oraz narzędzia edycji. Każdy z obiektów jest reprezentowany zbiorem definiowalnych atrybutów, takich jak kolor, font, rozmiar, prędkość animacji i inne. W każdym obiekcie można wyróżnić kilka warstw taktowanych komendą *naciśnij klawisz*. Formy graficzne mogą być uzupełnione elementami animacji i elementem interakcji. Opcja *EDYTOR* jest podstawowym elementem projektowania materiałów CAI z edytorem grafiki i animacji, edytorem tekstów, modułem animacji, edytorem dźwięków, programatorem funkcji (dla prostych obliczeń) oraz debugerem opracowanego obiektu.

Wysoki poziom programowania

Jednostki zostają złożone w program za pomocą konsolidatora pakietu. Złożenie tworzy menu odpowiadające nazwom obiektów. W etapie konsolidacji określamy dodatkowe elementy interakcji. Na zakończenie każdego ekranu można wywołać ekran następny lub klawiszem *Esc* – menu zawierające listę nazw ekranów. Odpowiednia manipulacja konsolidatorem zapewnia prostą interakcję w ramach opracowanego bloku programowego. W pakiecie ASYS 3.0 nie przewidziano narzędzi oceny odpowiedzi zaawansowanej interakcji.

Wymagania sprzętowe:
Proste PC – 12 MHz XT lub AT, 640 kB RAM, nie jest konieczny dysk twardy, karta z trybem EGA oraz DOS 3.x.

Charakterystyka APW

Authorware Professional for Windows 3.0 *Built* 18, jest wersją narzędzi autorskich przeniesionych z systemu MacIntosh [1]. Oprogramowanie to jest instalowane w środowisku Windows Microsoft na komputerach osobistych klasy IBM PC 286/386. Pełne wykorzystanie jego możliwości jest uwarunkowane zainstalowaniem dodatkowych kart i dodatkowego oprogramowania (edytory: *Write, Paintbrush, CorellDraw* itp.).

Instalacja APW

Pakiet APW może pracować na komputerach PC/AT 286/ /386 z dowolną kartą graficzną, z jednym dyskiem elastycznym i dyskiem twardym. Minimalna pojemność pamięci operacyjnej wynosi 1 MB. Praca z komputerem w takiej konfiguracji jest możliwa, choć bardzo mało efektywna, ponieważ jest brak możliwości wykorzystania niektórych opcji tego pakietu (w tym digitalizacji dźwięku i sterowania magnetowidem).

Właściwy komfort pracy zapewnia komputer z procesorem 80386 o częstotliwości zegara 33 MHz. Komputer powinien mieć monitor kolorowy z kartą graficzną VGA o rozdzielczości

600 × 800 punktów oraz możliwością 256 barw (środowisko okien graficznych zależne od parametrów komputera). Pojemność pamięci operacyjnej w istotny sposób ogranicza efektywność aplikacji APW. Pełne wykorzystanie trybu okien graficznych wymaga pamięci RAM o pojemności powyżej 640 kB. Słabą stroną wszystkich aplikacji w środowisku Windows 3.0 jest duża zajętość pamięci operacyjnej oraz wolna współpraca z dyskiem twardym. Zadowalający komfort pracy w środowisku APW zapewnia dysk twardy o pojemności nie mniejszej od 40 MB i szybkość transmisji danych (bez zewnętrznego bufora typu *cache*) nie mniejsza niż 500 kB/s. Znaczną poprawę parametrów transmisyjnych zapewnia bufor typu *cache* dysku twardego instalowany w pamięci operacyjnej komputera. Stąd pożądaną pojemnością pamięci operacyjnej komputera jest 4 MB.

Do wykorzystania możliwości dźwiękowych może służyć karta *Soundblaster*, pod warunkiem, że w komputerze jest wolne przerwanie sprzętowe Intr-7 (zwykle praca portu LPT1 w trybie „drukowania w tle") oraz adres 200 w przestrzeni adresowej we-wy. W pliku *win.ini* należy wpisać nazwę sterownika programowego, który obsługuje tę kartę.

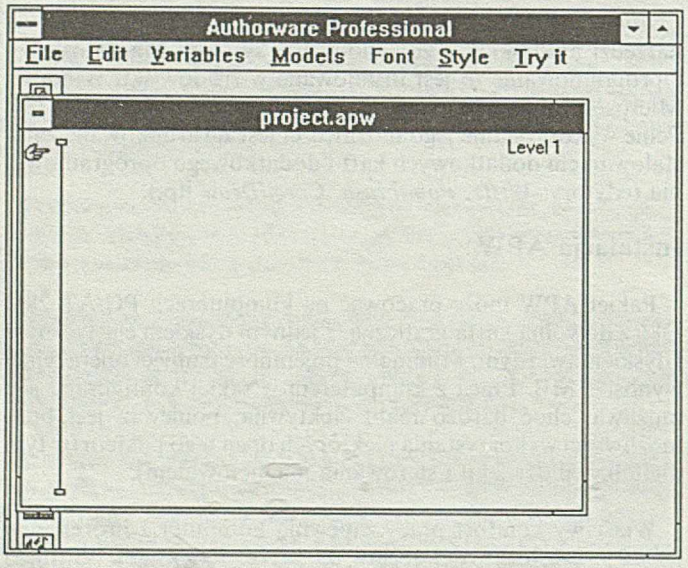
Pakiet APW umożliwia współpracę z jednym z trzech typów magnetowidów: firmy Pioneer (LDV-4200, LDV-6000) lub firmy Sony (rodzina LDP).

Instalacja pakietu APW odbywa się za pomocą programu zarządzającego (*File Manager*) plikami w środowisku Windows poleceniem *A: install*. Program instalacyjny kopiuje pliki z dwóch dyskietek dystrybucyjnych do odpowiednich katalogów na dysku twardym i umieszcza grupę ikon *Authorware* w środowisku okien systemu Windows 3.0.

Wszystkie wymagania programowe i sprzętowe dotyczą w równej mierze pisania aplikacji pod APW jak i uruchamiania gotowej aplikacji.

Podstawowe funkcje pakietu APW

Pakiet APW jest aplikacją środowiska Windows 3.0. Z tego też powodu wybór funkcji APW odbywa się w sposób charakterystyczny dla tego środowiska. Oznacza to, że APW jest wyposażony w menu, w którym poszczególne opcje są wybierane za pomocą myszy. W dalszej części artykułu przyjęto założenie, że Czytelnikowi znane jest środowisko systemu MS Windows i dlatego omówiono jedynie polecenia charakterystyczne dla APW. Polecenia typowe dla MS Windows zostały jedynie wymienione. Na rys. 1 pokazano elementy nierozwiniętego podstawowego menu systemu APW.



Rys. 1. Elementy nierozwiniętego systemu menu

Grupa poleceń *File*
Zawiera polecenia:
- tworzenia nowego projektu – *New file*,
- czytania z dysku projektu wcześniej utworzonego – *Open file*.

Istnieje możliwość importu grafiki w formatach *.WFM, .EPS, .BMP, .DIB, .TIF, .PIC*. Z aplikacji zrealizowanych w MS Windows-PaintBrush (najczęściej) są importowane pliki w formacie *.BMP*.

Grupa poleceń *Edit*
Oprócz podstawowych poleceń edytora graficznego i tekstowego, takich jak kopiowanie, kasowanie, wpisywanie do Clipboard-Windows, grupa ta zawiera polecenia charakterystyczne dla opisywanego pakietu. Opisane w rozdziale „Ikony" grupowanie ikon do jednej ikony grupującej polecenia to „Group" oraz „Ungroup". Polecenie „Effect" umożliwia rysowanie i kasowanie obiektów przez rozwijanie w sposób określony przez użytkownika. Opcje: *Lines, Fills, Modes* to odpowiednio: linie, wypełnianie środków i tryby rysowania określają grubości i zakończenia w rysowanych linii, wzory do wypełniania figur geometrycznych i sposoby umieszczania tych figur (np. inwersja, „przezroczyste" itp.).

Grupa poleceń *Variables*
Ogólnie można stwierdzić, że polecenia te umożliwiają przegląd i ewentualne dołączanie do projektu zmiennych oraz funkcji użytkownika.

Grupa poleceń *Models*
Jest charakterystyczna i unikalna dla pakietu APW. Polecenia umożliwiają tworzenie tzw. modeli – odpowiedników wcześniej opisanych ikon grupujących. Modele te, po nadaniu im nazwy i zapisaniu na dysku, można wykorzystywać w innych projektach (podobnie jak obiekty). Stwarza to możliwość tworzenia bibliotek modeli stosowanych w różnych projektach. Nazwy modeli mają rozszerzenie *.MOD*.

Grupa poleceń *Fonts*
Jest typowym zestawem opcji charakterystycznych dla edytora tekstu pracującego w trybie graficznym. Użytkownik ma do dyspozycji kilka rodzajów fontów z możliwością użycia ich w różnych skalach – liczonych w punktach drukarskich od 8 do 24 oraz w różnych stylach (pogrubienie, podkreślenie itp.).

Grupa poleceń *Tryit*
Dotyczy uruchamiania całości lub wybranego fragmentu projektu. Jest to narzędzie uruchomieniowe zaprojektowanej aplikacji. Po ewentualnym przerwaniu wykonywania programu, użytkownik dysponuje grupami poleceń dostosowanych do fazy programu, w której wystąpił błąd. Po poprawieniu błędu istnieje możliwość jego kontynuacji – od miejsca przerwania lub od początku. Można zauważyć istotny brak w tej grupie, a mianowicie automatycznego zapamiętywania źródła, co w momencie wystąpienia błędu „zawieszającego" pracę systemu Windows lub DOS powoduje utratę redagowanego projektu.

Sieć działań programu i przygotowanie aplikacji

Pakiet APW posługuje się niekonwencjonalnym zapisem tworzonych aplikacji. Praca w tym środowisku polega na edycji sieci działań – algorytmu projektowanego programu. Sieć

działań jest budowana z ikon opisanych w następnym rozdziale i łączonych ze sobą w łańcuch (ang. *flowline*), wyznaczający drogę realizacji programu (algorytmu). Zasadniczą treścią sieci działań pakietu jest grafika prezentacyjna z dodatkowymi efektami audiowizualnymi.

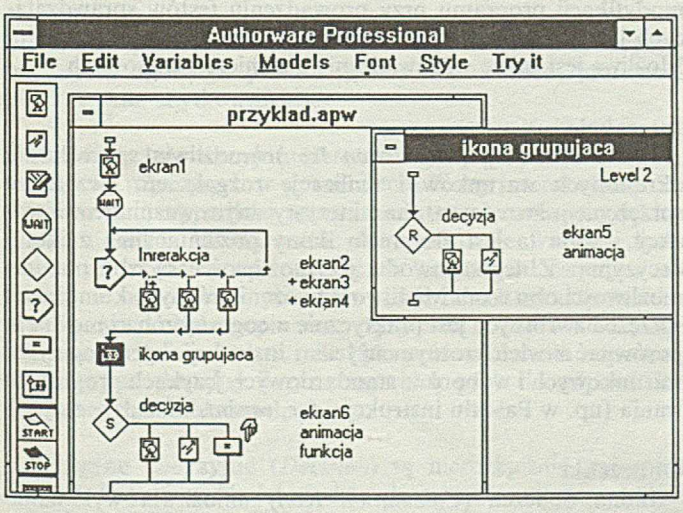
Tworząc nową aplikację na ekranie pojawia się pionowa linia reprezentująca główną linię programu (*flowline*). Linia ta jest ograniczona znacznikami początku i końca aplikacji. „Pisanie" programu rozpoczynamy od umieszczenia odpowiednich ikon na głównej linii sieci działań programu (wybranych z palety ikon). Wpisana ikona jest natychmiast aktywna, ale nie ma tytułu. W trakcie nadawania ikonie tytułu deklarujemy jej dodatkowe parametry.

Dwie ikony umożliwiają tworzenie rozgałęzień w programie. Są to ikony: decyzyjna i interakcyjna. Wykorzystując te ikony możemy wybrać jedną z możliwych dróg, w zależności od spełnionych warunków. Ikona interakcyjna jest odpowiednikiem instrukcji warunkowej i pętli programowej w konwencjonalnych językach programowania. Ikona decyzyjna jest odpowiednikiem instrukcji wyboru i sekwencyjnych pętli programowych.

Po zaprojektowaniu fragmentu sieci działań istnieje możliwość zgrupowania tej części programu w jednej ikonie grupującej lub zdefiniowania modelu, który reprezentuje ten fragment sieci działań. Różnica między oboma sposobami zapisu polega na tym, że modele można kopiować do innych aplikacji, natomiast ikonę grupującą używamy w aplikacji bieżącej. Ta technika projektowania aplikacji stwarza możliwość stworzenia stosunkowo dużych programów w przejrzystej formie. Częściowo zastępuje to procedury występujące w konwencjonalnych językach programowania. Nie ma w tej opcji możliwości przekazywania parametrów formalnych.

Aplikacja może być wielopoziomowa. Główny poziom programu zawiera ikony grupujące, które są opisane w niższych poziomach aplikacji. Poszczególne poziomy sieci działań są umieszczane w indywidualnych okienkach środowiska Windows, zatytułowanych jak odpowiadająca im ikona grupująca. Okno takie jest obsługiwane podobnie jak główny poziom aplikacji.

Środowisko APW zawiera program uruchomieniowy (*debugger*), umożliwiający krokowe wykonywanie programu i usuwanie ewentualnych błędów. Do zaznaczania początku i końca uruchomianego fragmentu sieci działań służą dwie ikony: początkowa – startowa oraz końcowa. Ponadto, jeśli program uruchamiający natrafi na błędnie zredagowaną ikonę w sieci działań, następuje automatyczne przejście do trybu edycji tej ikony. Poprawianie projektu polega na przeredagowaniu błędnych parametrów ikony. Można też kasować i przenosić w inne miejsce sieci działań ikony istniejące oraz dopisywać nowe.



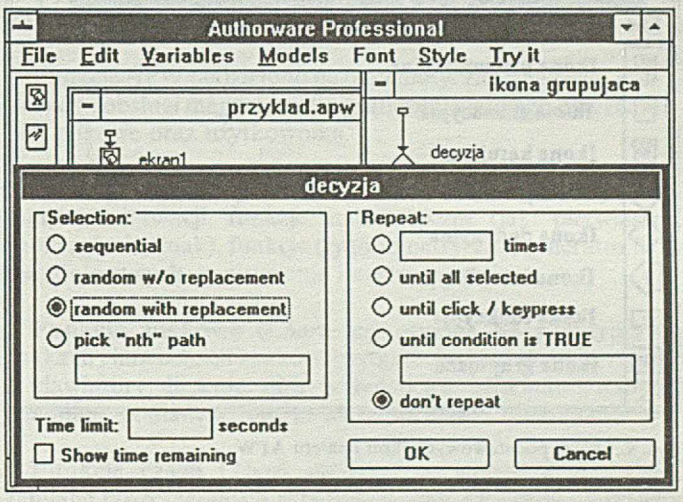
Rys. 2. Ekran roboczy środowiska APW

Powyższe omówienie zilustrowano przykładem projektu pokazanym na rys. 2. Ekran roboczy środowiska APW podzielono na trzy części. W górnej części umieszczono rozwijane menu (opisane wyżej), natomiast w części lewej ekranu – zestaw ikon. Pozostała część ekranu to obszar roboczy, na którym projektant redaguje sieć działań.

Przykładowa aplikacja składa się z dwóch poziomów. Poziom drugi reprezentuje wyróżniona ikona grupująca. Stanowi ona podprogram, w którym losowo (*R – Randomize*) zostaje podjęte działanie ikony o nazwie *ekran*6 lub ikony o nazwie *animacja*. Reakcję tę określa ikona decyzyjna oznaczona symbolem *R*.

Najniższy poziom przygotowania projektu, to edycja parametrów wszystkich wybranych ikon. Na rys. 3 przedstawiono przykład ekranu edycyjnego ikony decyzyjnej *Decyzja*. Parametrami, które możemy zadeklarować (dla tej ikony) to:

- rodzaj wyboru,
- rodzaj powtarzania,
- limit czasu na podjęcie decyzji.



Rys. 3. Ekran edycyjny ikony decyzyjnej

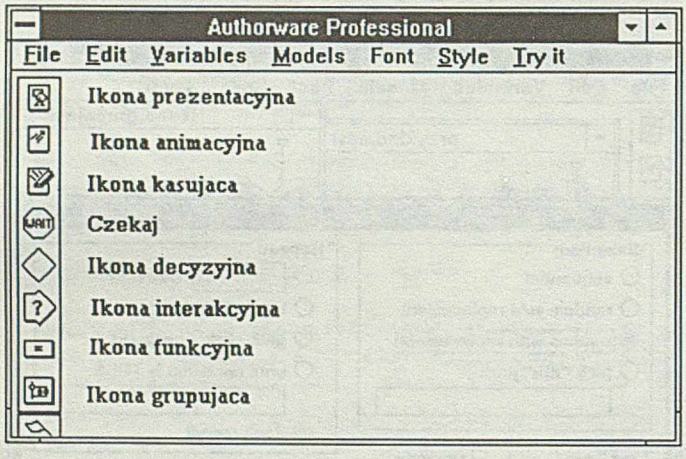
Wybór decyzji może być sekwencyjny, pseudoprzypadkowy z pojedynczym i wielokrotnym wyborem oraz z wyborem określonego numeru ścieżki. Powtarzanie może następować: określoną liczbę razy aż do wybrania wszystkich ikon, do chwili naciśnięcia przycisku myszy albo klawiatury lub tak długo, jak długo jest spełniony określony warunek. Limit czasowy może być określony w sekundach z możliwością wyświetlania na ekranie upływającego czasu.

Poziom pierwszy projektu składa się z kilku ikon reprezentujących wyświetlanie ekranów od 1 do 6 według zadanego algorytmu. Ekran1 jest wyświetlany bezwarunkowo, zawsze po rozpoczęciu realizacji programu, po czym następuje zatrzymanie programu i oczekiwanie na określoną akcję: naciśnięcie wybranego klawisza klawiatury lub przycisku myszy, względnie upływ zadanego czasu. Kolejne trzy ekrany są wybierane zależnie od kodu wprowadzonego przez użytkownika – analizo-

wanego w ikonie *interakcja*. Działania *ikony grupującej* opisano wcześniej. Program kończy sekwencyjne wykonanie ikon: *Ekran 6*, *Animacja* i *Funkcje*. Tak zapisany projekt stanowi program źródłowy, który można zapisać do pliku z rozszerzeniem **.apw*. Taki plik można czytać i uruchamiać w środowisku APW lub za pomocą zewnętrznego interpretera *ApwRun*. Projekt kończymy operacją tworzenia pliku z **.exe*, który można uruchamiać w środowisku Windows jako jego aplikację. Do przygotowania takiego pakietu służy opcja *Package* w menu *File*.

Ikony

Projekt programu w systemie APW jest tworzony za pomocą ikon (ang. *Design Icons*). Ikony są blokami algorytmu zapisanego w formie sieci działań. Funkcje tych ikon różnią się od konwencjonalnie przyjętego sposobu zapisu algorytmów w postaci graficznej. Generalnie można stwierdzić, że ikony cechują większe możliwości od pojedynczych bloków algorytmów i zastępują często kilka lub kilkanaście instrukcji konwencjonalnego języka programowania. Poniżej przedstawiono charakterystykę ikon jako narzędzia projektowania (pisania) programu użytkowego. Na rys. 4 przedstawiono osiem podstawowych ikon edytora APW. Każda ikona ma swój unikalny symbol graficzny, reprezentujący określone funkcje, które w określonym zakresie można modyfikować.



Rys. 4. Osiem podstawowych ikon pakietu APW

Poniżej przedstawiono charakterystykę ikon z tzw. zestawu podstawowego.

Prezentacja tekstu i grafiki (*Display Icon*)

Projektowanie grafiki i tekstu w środowisku APW umożliwiają podstawowe – niestety dość ubogie narzędzia, takie jak rysowanie linii i prostych figur geometrycznych wypełnianych kolorami – w trybie standardowym lub transparentnym. Grafika może być uzupełniana tekstem o zadowalającej liczbie fontów (kilka rozmiarów i krojów pisma). Istnieje jednak możliwość zewnętrznego importu grafiki tekstu do projektowanej aplikacji. W wersji APW – *build* 18 możliwości importu są jednak ograniczone do plików ASCII i obrazów typu *mapa bitów*. Istnieje także możliwość nałożenia kolejnych prezentacji graficznych z kilku ikon.

Podsumowując, prezentacja tekstu i grafiki zawiera wszystkie typowe narzędzia edytora tekstowo-graficznego. Jednak operowanie tymi narzędziami wymaga dość dużej wprawy i czasem kończy się niezamierzonym efektem, co znacznie wydłuża przygotowanie programu. Ikona *Display* jest jednym ze słabszych ogniw pakietu APW. Jej możliwości byłyby znacznie większe przy możliwości importu obiektów z innych, bardziej rozbudowanych narzędzi tekstowo-graficznych, jak np. *CorelDraw*. W ikonie Display, podobnie jak w większości innych, można wykorzystywać niżej omówione tzw. zmienne systemowe.

Animacja

Ikona animacji (*Animation Icon*) umożliwia przesuwanie obiektu lub grupy obiektów na ekranie użytkownika z określoną prędkością lub w określonym czasie. Tor przesuwania (animacji) obiektu można określić na kilka sposobów:

- tor od aktualnej pozycji do pozycji docelowej,
- tor określony drogą,
- tor określony jak wyżej, lecz z możliwością powrotów lub zatrzymań w zależności od ustalonych wartości zmiennych systemowych,
- tor określony linią prostą,
- tor określony prostokątnym układem współrzędnych.

Istotną cechą animacji w APW jest możliwość realizacji ruchu kolejnych obiektów (cyklicznie), co stwarza wrażenie animacji równoległej.

Kasowanie

Ikona kasowania (*Erease Icon*) służy do kasowania wyświetlanych obiektów tekstowych i graficznych. Proces kasowania można modyfikować, co oznacza możliwość kasowania „rastrowego": od środka ekranu na zewnątrz, z góry na dół, z prawej do lewej itp. Kasowanie takie nie powoduje przykrego dla użytkownika efektu nagłych błysków, czy zaciemnień ekranu.

Czekanie

Ikona czekania (*Wait Icon*) przerywa działanie programu na określony czas lub do momentu naciśnięcia dowolnego klawisza klawiatury lub przycisku myszy. Dodatkowo można definiować klawisze przerywające fazę czekania z możliwością wyświetlenia upływającego czasu (stopera).

Decyzja

Ikona decyzji (*Decision Icon*) pozwala na wybór następnej ikony z zestawu ikon dostępnych w bieżącej fazie realizacji programu. Wyboru można dokonać w sposób pseudolosowy lub określony przez użytkownika, np. wybór drogi w sieci działań programu uzależniony od poprzednich rezultatów i wyników obliczeń arytmetycznych. Ma to szczególne znaczenie dla modyfikacji programu przy prowadzeniu testów sprawdzających lub oceniających poznanie poprzednich partii materiału. Możliwe jest także wprowadzenie ograniczeń czasowych.

Interakcja

Ikona interakcji (*Interaction Icon*) umożliwia sprawdzenie określonych warunków i realizację rozgałęzień, łącznie ze sprzężeniem zwrotnym, charakterystycznym warunkiem interakcji. Ikona ta jest złożeniem ikony prezentacyjnej z ikoną decyzyjną. Z tego powodu jej możliwości są zbliżone do możliwości obu ikon. Możliwość twożenia różnych kombinacji sprzężeń zwrotnych jest praktycznie nieograniczona i można ją porównać z wielokrotnym użyciem instrukcji interakcyjnych, warunkowych i wyboru w standardowych językach programowania (np. w Pascalu instrukcje: *for, repeat, while, case*).

Obliczenia

Ikona obliczeń (*Calculation Icon*) umożliwia wykonanie podstawowych obliczeń arytmetycznych, wywołanie funkcji

użytkownika oraz wykonanie skoków do innej aplikacji. Wartości zmiennych uzyskane z tych obliczeń można wykorzystywać w innych ikonach, np. do skalowania wykresów lub wyboru dróg w ikonach decyzyjnych czy interakcyjnych.

Grupowanie

Ikona grupowania (*Map Icon*) grupuje wyróżnioną część lub całość sieci działań i umieszcza wszystkie te ikony oraz funkcje, które są przez nie wykonywane w jednej ikonie – ikonie zastępczej: grupującej. Może ona zawierać także inne ikony grupujące. Taka technika tworzenia aplikacji przypomina strukturalne tworzenie programu.

Wskaźniki początku i końca

Znaczniki początku i końca (*Start/Stop – Flags*) umieszczone w wybranych punktach sieci działań umożliwiają realizację wybranych fragmentów programu: od znacznika początku do znacznika końca.

Obsługa narzędzi multimedialnych APW jest realizowana za pomocą omówionych poniżej ikon uzupełniających. Uruchomienie tej części oprogramowania jest możliwe jedynie wtedy, gdy zostały zainstalowane odpowiednie karty sterujące do komputera oraz sterowników programowych środowiska Windows.

Magnetowid

Ikona wideo (*Video Icon*) umożliwia sterowanie funkcjami magnetowidu na zasadach pilota zdalnego sterowania. Program użytkownika włącza magnetowid stosownie do zapisanego scenariusza.

Dźwięk

Ikona dźwięku (*Sound Icon*) umożliwia zapisanie do pamięci komputera, a następnie odtworzenie „zdigitalizowanego" dźwięku w formacie PCM – modulacja impulsowo-kodowa.

Ruchomy obraz

Ikona ruchomego obrazu (*Movies Icon*) umożliwia uruchomienie specjalnej stacji dyskowej z ruchomymi obrazami zapisanymi w plikach PICS. Technika ta jest związana z aplikacjami w systemach Macintosh i nie wydaje się, aby obecnie była dostępna na komputerach PC klasy IBM.

Zmienne i funkcje

Pakiet APW udostępnia dość szeroki wybór zmiennych i funkcji, które można wykorzystać w programie do przechowywania interakcji. Są one uzupełnieniem technik opisanych wcześniej przy okazji omawiania ikon decyzyjnych, interakcyjnych i obliczeniowych.

Zmienne pakietu APW podzielono na interakcyjne, decyzyjne, czasu, ogólnego przeznaczenia, stanu wideo, graficzne, plikowe oraz użytkownika.

Zmienne interakcyjne (*Interaction*) są grupą zmiennych, najczęściej używanych w uzupełnieniu ikony interakcyjnej. Są w większości przypadków ustawiane automatycznie przy deklarowaniu reakcji użytkownika. Na podstawie wartości tych zmiennych możemy np. określić, ile czasu miał ćwiczący na udzielenie odpowiedzi, po jakim czasie udzielono odpowiedź, czy odpowiedź była poprawna, czy pierwsza próba udzielenia odpowiedzi była poprawna, ile było prób udzielenia odpowiedzi, jaki jest aktualny iloraz wszystkich prób do odpowiedzi poprawnych lub błędnych.

Zmienne decyzyjne (*Decision*) są nierozłącznie związane z ikoną decyzyjną. Na ich podstawie można stwierdzić, która z możliwych dróg została obecnie wybrana, czy wszystkie z możliwych dróg były wcześniej wybrane, czy jest to pierwszy wybór, ile już razy wybrano aktualną drogę, ogólną liczbę powtórzeń.

Zmienne czasu (*Time*), których wartości reprezentują datę i czas systemowy oraz poszczególne ich wycinki (dzień, minuty, nazwa dnia i tygodnia itp.), datę i czas pierwszej sesji z aplikacją APW, czas który upłynął od początku sesji.

Zmienne ogólnego przeznaczenia (*General*) zawierają informację o uruchomionym systemie operacyjnym i wersji APW, aktualne położenie kursora myszy, informacje o stanie klawiatury, informacje o aktualnie wyświetlanym oknie, kod powrotu z programu.

Zmienne stanu wideo (*Video*) informują o stanie sygnałów sterujących magnetowidu.

Zmienne graficzne (*Graphics*) reprezentują ostatnie aktywne położenie kursora graficznego.

Zmienne plikowe (*File*) zaierają informacje o aktualnie wybranym pliku, dysku i statusie zakończonej operacji we-wy.

Zmienne użytkownika (*User*) są grupą zmiennych deklarowanych przez użytkownika. Zmienne te można w prosty sposób zadeklarować z poziomu menu *Variables* w opcji *New*. Różnica między zmiennymi systemowymi i użytkownika polega przede wszystkim na tym, że zmienne systemowe podlegają automatycznej modyfikacji, natomiast o wartości zmiennych użytkownika decyduje piszący program.

Funkcje APW podzielono na matematyczne, znakowe, czasu, skoków, obsługi magnetowidu, graficzne, ogólnego przeznaczenia, plikowe oraz użytkownika.

Funkcje matematyczne (*Math*) zawierają operatory matematyczne i relacji, funkcje matematyczne (jak: pierwiastek, zaokrąglenie, znak), funkcje trygonometryczne, generator liczb pseudolosowych, zapisywanie i czytanie tablic.

Funkcje znakowe (*Character*) operują na pojedynczych znakach i łańcuchach znaków: czytanie znaku, łańcucha i liczby z klawiatury, liczenie słów w łańcuchu, szukanie-wycinanie i zamiana podłańcuchów, zamiana na duże i małe litery.

Funkcje czasu (*Time*) umożliwiają wprowadzanie daty, zamianę poszczególnych składników daty na liczby i łańcuchy reprezentujące ich nazwy.

Funkcje skoków (*Jump*) umożliwiają skok w dowolne miejsce aplikacji (do dowolnego ekranu), wykonanie zewnętrznego pliku programowego, zakończenie wykonywania programu z określonym kodem powrotu.

Funkcje obsługi magnetowidu (*Video*) umożliwiają włączenie dźwięku i obrazu oraz zmianę prędkości pracy magnetowidu.

Funkcje graficzne (*Graphics*) umożliwiają narysowanie kółka, linii i prostokąta o zadanych rozmiarach.

Funkcje ogólnego przeznaczenia (*General*) umożliwiają zmianę rozmiaru lub „zwinięcie" do ikony aktywnego okna, ustawienie lub zmianę kształtu, zgaszenie i zapalenie kursora, użycie dźwiękowego sygnału ostrzegawczego, wydrukowanie aktualnego ekranu na drukarce (ang. *print screen*), zapisanie rekordu użytkownika na dysku, zakończenie wykonywania programu.

Funkcje plikowe (*File*) umożliwiają sprawdzenie zawartości katalogu na dysku, utworzenie i skasowanie pliku lub podkatalogu, zapisanie i odczyt łańcucha znakowego z pliku na dysku, określenie typu wybranego pliku.

Funkcje użytkownika (*User*) są przygotowywane przez użytkownika w innym środowisku programowym, dołączane i wykorzystywane do pisania aplikacji pod kontrolą APW. Funkcje, które mają być dołączone muszą być zapisane w pliku o formacie DLL (*Microsoft Dynamic Link Libraries*). Plik taki stanowi wstępnie skompilowane funkcje, które mogą być uruchomione z poziomu aplikacji działających pod nadzorem systemu Windows. Do niedawna jedynym sposobem otrzymania pliku w formacie DLL, było napisanie funkcji w języku MS C wersja 5.1 lub późniejsza, skompilowanie jej kompilatorem z tego pakietu, a następnie utworzenie pliku DLL przy użyciu oprogramowania *Software Development Kit for Windows* 3.0. Wymagało to użycia kosztownego oprogramowania i poświęcenia znacznego czasu na naukę skomplikowanego środowiska programowego oraz napisania i przygotowanie w odpowiednim formacie (DLL) funkcji. Sytuację poprawiło wprowadzenie na rynek przez firmę Borland International pakietu Pascal for Windows 3.0, za pomocą którego można przygotować funkcję w jednym przebiegu kompilacji. Nie bez znaczenia jest też fakt, że środowisko programowe firmy Borland International jest ponad 10 razy tańsze od środowiska firmy Microsoft. Można zatem uznać, że użytkownik dysponujący pakietem Pascal for Windows i umiejący go obsługiwać może stworzyć dowolne funkcje, które nie są standardowo dostępne z pakietu APW. W ten sposób można wzbogacać i rozbudowywać aplikacje napisane w APW o dowolne elementy.

* * *

Z omówionych wyżej trzech rodzajów oprogramowania autorskiego każde ma różne walory. SHIVA jest bardzo zachęcające z uwagi na zintegrowaną formę. Ma mniejsze wymagania sprzętowe i systemowe w zakresie uzupełniającego środowiska jak Windows, czy importery graficzne i dodatkowe narzędzia graficzne, występujące w przypadku APW. Pakiet APW zachęca ogromnymi możliwościami, ale można je wykorzystać tylko na komputerze wysokiej klasy oraz przy bardzo złożonym i kosztownym oprogramowaniu uzupełniającym. Ponadto instalacja APW, stopień złożoności i wysokie koszty przedsięwzięcia nie są w chwili obecnej zachęcające. ASYS z punktu widzenia możliwości redakcyjnych nie może niestety konkurować z żadnym z wymienionych wyżej opracowań. Można jednak stwierdzić, że pokrywa on znaczną część najważniejszych narzędzi niezbędnych do produkcji materiałówCAI z prostą warstwą interakcji. Nie bez znaczenia jest tu również niska cena wyposażenia sprzętowego i programowego.

LITERATURA

[1] Authorware Professional for Windows 3.0 build 18, User's Manual. Authorware Europe, 1991

[2] Delta Project D1010: Advanced Authoring Tools. IRPEACS laboratory of the CNRS. Lyon 1990, France

[3] Piecha J.: The multilevel model for microcomputer ICAI systems. The International Journal of Applied Engineering Education, Vol. 5, No 3/89. Pergammon Press, Oxford 1989

[4] Piecha J. i inni: System autorski ASYS. Funkcje i zasady użytkowania. Komputery w Dydaktyce. Wydawnictwa IKZ, Min. E. N., s. 30–40, Warszawa 1990

[5] Piecha J.: Podstawy projektowania systemów CAI. Informatyka s. 1–6, 4, 1991

[6] Piecha J.: Proc. of Int. Conference on Computer Aided Learning and Simulation Technologies. Unesco Centre, Prague, June 1991.

Inteligentny system gromadzenia informacji

dokończenie ze s. 13

Niniejszy artykuł zawiera pierwszą część wyników pracy nad zagadnieniem informacji faktograficznej oraz nad metodami jej gromadzenia i przetwarzania. Z informacją faktograficzną spotykamy się bardzo często, równie częsta jest potrzeba jej dokumentowania. Zbiory FG mogą być podstawą dla różnego rodzaju procesów wnioskowania, stąd też pomysł uporządkowania definicji oraz propozycja języka i systemu do automatycznego gromadzenia ciągów FG. Formułując prezentowany tu system gromadzenia informacji FG wybrano postępowanie, polegające na tworzeniu repliki sytuacji problemowej w pamięci systemu i repliki metod działania. Koncepcja taka, zapożyczona z badań nad systemami sztucznej inteligencji, jest uzasadniona złożonością sytuacji problemowych. Innym elementem koncepcji utworzonego tu systemu GFG jest wyrażenie działań systemu w postaci procesów dialogowych, z przypisaniem roli wiodącej systemowi. Raz wprowadzona do systemu wiedza o sytuacji i metodzie gromadzenia lub szukania uwalnia użytkownika systemu od trudów podejmowania decyzji, a wyniki, tj. informację FG zgromadzoną lub/i przetworzoną, od błędów użytkownika spowodowanych np. przez spadek koncentracji.

Niniejszy artykuł zawiera także pierwszą część wyników pracy nad systemem GFG obejmującą mechanizmy gromadzenia pierwotnego ciągu FG. System dysponuje również mechanizmami do przeszukiwania ciągu pierwotnego, jest to jednak temat bardzo obszerny i wymaga osobnego omówienia.

LITERATURA

[1] Barr A., Feigenbaum E.: The Handbook of Artificial Intelligence. Vol. 1., W. Kaufmann Inc., Los Altos, Ca..1981

[2] Deininger R.A.. (Ed.): Design of Environmental Information Systems, Ann Arbor, Michigan, 1974

[3] Firebaugh M.W.: Artificial Intelligence – Knowledge-Based Approach. Boyd & Fraser Publ. Co., Boston 1988

[4] McCarthy J. et al.: LISP 1.5 Programmer's Manual. MIT Press, Cambridge. MA, 1962

[5] Minsky M.: Framework for representing knowledge. W pracy: Winston H. (Ed.): The Psychology of Computer Vision. McGraw Hill, 1975

[6] Steele G.L.: Common LISP – The Language. Digital Press, Burlington, MA, 1984

[7] Wilson A.L.: The design of long term programs for the measurement of river water quality. W publikacji [2]

[8] Wrzeszcz Z.: Opracowanie algorytmów systemu gromadzenia i wyszukiwania informacji faktograficznej z wykorzystaniem metod sztucznej inteligencji. IMM. Raport badawczy, Warszawa 1989

[9] Wrzeszcz Z.: System z bazą wiedzy do gromadzenia informacji faktograficznej. „Inżynieria Wiedzy i Systemy Ekspertowe", I Krajowa Konferencja Naukowa, Wrocław, 26–28.06.1990., Politechnika Wrocławska 1990

[10] Wrzeszcz Z.: Język Lisp i elementy sztucznej inteligencji. INFORMATYKA, nr 9, 1990.

Polskie Towarzystwo Informatyczne Oddział Górnośląski zaprasza do udziału w

IV Wyższej Górskiej Międzynarodowej Szkole PTI pn. „Współczesne obszary zastosowań informatyki",

która odbędzie się w okresie 22–26 czerwca 1992 r. w Szczyrku w domu wczasowym „Klimczok".

Program Szkoły obejmuje różne formy przekazywania informacji, takie jak: wykłady, spotkania autorskie, warsztaty, gry symulacyjne i inne.

Głównym jej tematem jest komputerowe wspomaganie decyzji. Przewidziane są wykłady dotyczące konstrukcji i implementacji tekstów pozwalających ocenić spełnienie wymagań metodycznych w szkole, prezentacje programu wspomagającego zapamiętywanie informacji, omówienie znaczenia komputerowych gier symulacyjnych w kształceniu menedżerów oraz prezentacja metodyki modelowania systemów decyzyjnych.

Organizatorzy: PTI, 40-014 Katowice, ul. Mariacka 6, tel. 53-81-02.

/mail

Jakim jesteś unixowcem?

Każdy człowiek działający w dowolnej dziedzinie jest poddawany ocenom innych ludzi. Mówi się *on jest dobry* albo *on się nie orientuje*. Ponieważ ocena zwykle zależy od kompetencji cenzora, więc grupy specjalistów jednej dziedziny lub pojedynczy eksperci opracowują kryteria. W liście p. Jacka Surmy publikującego swoje teksty na łamach „Informatyki" przeczytałam kryteria oceny unixowca udostępnione przez bezsprzecznego czarodzieja unixowego ze ścisłej czołówki – Gene Stafforda. Wyróżnia on i określa następujące stopnie wtajemniczenia:

1) początkujący
– nie jest pewien czym tak naprawdę jest terminal,
– nie nauczył się jeszcze podstaw edytora *vi*,
– nie wie jak wyświetlić zawartość katalogu,
– ciągle zapomina, że po wprowadzeniu każdego wiersza trzeba wcisnąć ⟨*Return*⟩;

2) nowicjusz
– wie już, że zawartość katalogu wyświetla się instrukcją *ls*,
– używa edytora *vi*, ale nazywa go *waj*,
– słyszał o *C*, ale nigdy go nie używał,
– miał pierwsze złe doświadczenia z instrukcją *rm*,
– zastanawia się jak przeczytać swoją pocztę,
– myśli – *dlaczego ten człowiek siedzący przy sąsiednim terminalu tak bardzo lubi UNIX?;*

3) użytkownik
– używa *vi* i *nroff*, ale nie jest ekspertem w tym zakresie,
– słyszał o wyrażeniach regularnych, ale nigdy żadnego nie widział,
– odkrył, że pauza (–) poprzedza opcje,
– usiłował napisać program w *C*, ale zdecydował się pozostać przy Pascalu,
– zastanawia się, jak przenieść katalog,
– sądzi, że termin *dbx* należy wiązać ze sprzętem stereo,
– wie, jak przeczytać swoje listy i zastanawia się, jak przeczytać nowinki (ang. *news*);

4) zaawansowany
– swobodnie używa *nroff* i zaczyna uczyć się *tbl* i *eqn*,
– używa *grep* do wyszukiwania wzorców tekstowych,
– odkrył, że *mv* przenosi także katalogi,
– stwierdził, że *learn* niczego nie uczy,
– ktoś mu pokazał jak pisać programy w *C*,
– użył raz *sed* do zamiany wzorców tekstowych,
– widział, jak ktoś używał *dbx*, ale sam tego nie próbował,
– uważa, że *make* jest używany tylko przez słabeuszy;

5) ekspert
– używa *sed* wtedy, gdy potrzeba,
– używa makrodefinicji w edytorach *vi* i *ex* wtedy, gdy potrzeba,
– wysyła nowinki przy każdej sposobności,
– od czasu do czasu pisze skrypty w *csh*,
– programy w *C* pisze używając edytora *vi* i kompiluje używając *cc*,
– odkrył po co są operatory && i !!,
– wierzy, że historia ludzkości zaczyna się od *!h*;

6) haker
– swobodnie używa *sed* i *awk*,
– korzysta z nigdzie nie opisanych możliwości *vi*,
– pisze programy w *C* używając instrukcji *cat* ⟩ i kompiluje je korzystając z *!cc*,
– używa *adb*, gdyż nie wierzy debugerom symbolicznym,
– umie odpowiedzieć na pytanie o środowisko użytkownika,
– pisze własne makrodefinicje dla *nroff* uzupełniając bibliotekę *std.ones*,
– pisze skrypty w shellu Bourne'a,
– wie, jak zainstalować poprawki systemowe;

7) guru
– swobodnie używa *m*4 i *lex*,
– pisze programy asemblerowe używając instrukcji *cat* ⟩,
– operuje na jądrze używając *adb* w czasie pracy systemu,
– poznaje lub modyfikuje programy narzędziowe czytając lub zmieniając ich kod źródłowy,
– przy śniadaniu czyta kody źródłowe driverów,
– po chwili zastanowienia odpowiada na dowolne pytanie o UNIX-ie,
– używa *make* zawsze, gdy uzyskanie czegoś wymaga użycia dwóch lub więcej oddzielnych komend,
– nauczył włamywać się do systemu, i więcej nie próbuje tego robić;

8) czarodziej
– pisze drivery urządzeń używając instrukcji *cat* ⟩,
– usuwa błędy modyfikując kody binarne,
– może odpowiedzieć na każde pytanie zanim zostanie ono wypowiedziane,
– pisze własne pakiety makrodefinicji dla *troff*,
– jest po imieniu z Dennisem, Billem i Kenem.

Sądzę, że tę klasyfikację należy traktować raczej żartobliwie, choć z pewnością każdy, kto ma do czynienia z UNIX-em spróbuje ocenić swój stopień zaawansowania.

AGNIESZKA PIETRUSIŃSKA

POLNET Technologies (członek organizacji UNIX INTERNATIONAL i organizator Polskiej Grupy Marketingowej Unix International) przy współudziale Centrum Promocji Informatyki FRTK organizuje kursy użytkowania, programowania i administracji systemem operacyjnym UNIX.

Program kursów i materiały szkoleniowe oparte są na doświadczeniach zdobytych przez POLNET TRAINING CENTER podczas szkoleń wielu różnych grup polskich użytkowników, programistów i administratorów systemu UNIX.

Każdy tygodniowy cykl szkoleniowy obejmuje wykład ilustrowany przykładami i ćwiczeniami praktycznymi ułatwiającymi zdobycie umiejętności praktycznych.

Informacje:
Centrum Promocji Informatyki FRTK
00-503 Warszawa, ul. Żurawia 4a
tel. 25-02-99, 25-60-59, 21-76-26

Informatyka w Singapurze

Singapur w ostatnich latach był symbolem ożywionego rozwoju gospodarczego, stabilności politycznej oraz ośrodkiem przyciągającym pracowników różnych narodowości i kapitał różnych firm. W 1990 r. obroty handlowe tego kraju osiągnęły poziom 120 mld dolarów, dochód na mieszkańca wyniósł 12 785 dolarów, a bezrobocie spadło do 1,7%. Ten szybki rozwój wymaga znacznego dopływu specjalistów z dziedziny techniki i mimo, że na oświatę przeznacza się tu aż 20% budżetu, konieczne jest zatrudnianie wielu cudzoziemców, zwłaszcza naukowców i inżynierów szczególnie w dziedzinie informatyki. Wydajność pracowników w Singapurze jest od wielu lat najwyższa w świecie wyprzedzając takie kraje, jak Szwajcaria i Japonia. Zagraniczne inwestycje w przemyśle wytwórczym Singapuru uchodzą za bardzo pewne, a ich udział przekracza obecnie 90%.

Już na początku lat osiemdziesiątych utworzono Krajową Radę Komputerową, co m.in. miało potwierdzić wysoką pozycję tego kraju w dziedzinie sprzętu i oprogramowania komputerowego oraz telekomunikacji. Wkrótce znane firmy zaczęły zakładać w tym kraju swoje oddziały. Pierwszą była Lotus Development Corp., która umieściła tu swe przedstawicielstwo na obszar Azji i Pacyfiku, a następnie utworzyła największy w tym kraju ośrodek naukowo-badawczy oprogramowania (powierzchnia 5 tys. m²). Również Apple Computer Inc. działa tu od 1981 r. Początkowo była to wytwórnia komputerów Apple II dla obszaru Pacyfiku i Stanów Zjednoczonych. W 1988 r. zakład został rozbudowany i obecnie wytwarza również inne modele komputera Macintosh, a wkrótce asortyment ma objąć komputery typu laptop. Zakład zatrudnia ponad 500 osób i pracuje na trzy zmiany.

Współpraca między Singapurem a USA układała się przez wiele lat z korzyścią dla obu stron. W październiku 1990 r. zostało podpisane porozumienie o handlu i inwestycjach, co w 1992 r. powinno zaowocować jeszcze lepszą współpracą. Dla poszczególnych firm oznacza to wzajemne stosowanie przepisów antydumpingowych, ochronę własności intelektualnej, a także ułatwienie dostępu do rynku towarów i usług.

Łączne inwestycje firm amerykańskich działających w Singapurze wyniosły w 1990 r. ok. 9 mld dolarów, co stanowiło 48% wszystkich zrealizowanych w tym kraju inwestycji zagranicznych. Koszt prowadzenia działalności jest tu dosyć wysoki, ale możliwości zysków w określonych dziedzinach są znaczne. Jako perspektywiczne dziedziny wymienia się technologię informacyjną, biologiczną i automatyzację. Pierwsza z tych dziedzin wymaga rozwoju komputerów, urządzeń peryferyjnych i oprogramowania użytkowego. Inne obszary szybkiego wzrostu to sprzęt telekomunikacyjny i usługi, a także sprzęt elektroniczny i jego elementy. Singapur jest również ważnym ośrodkiem naprawy i przeglądów samolotów oraz pomocniczego sprzętu lotniczego.

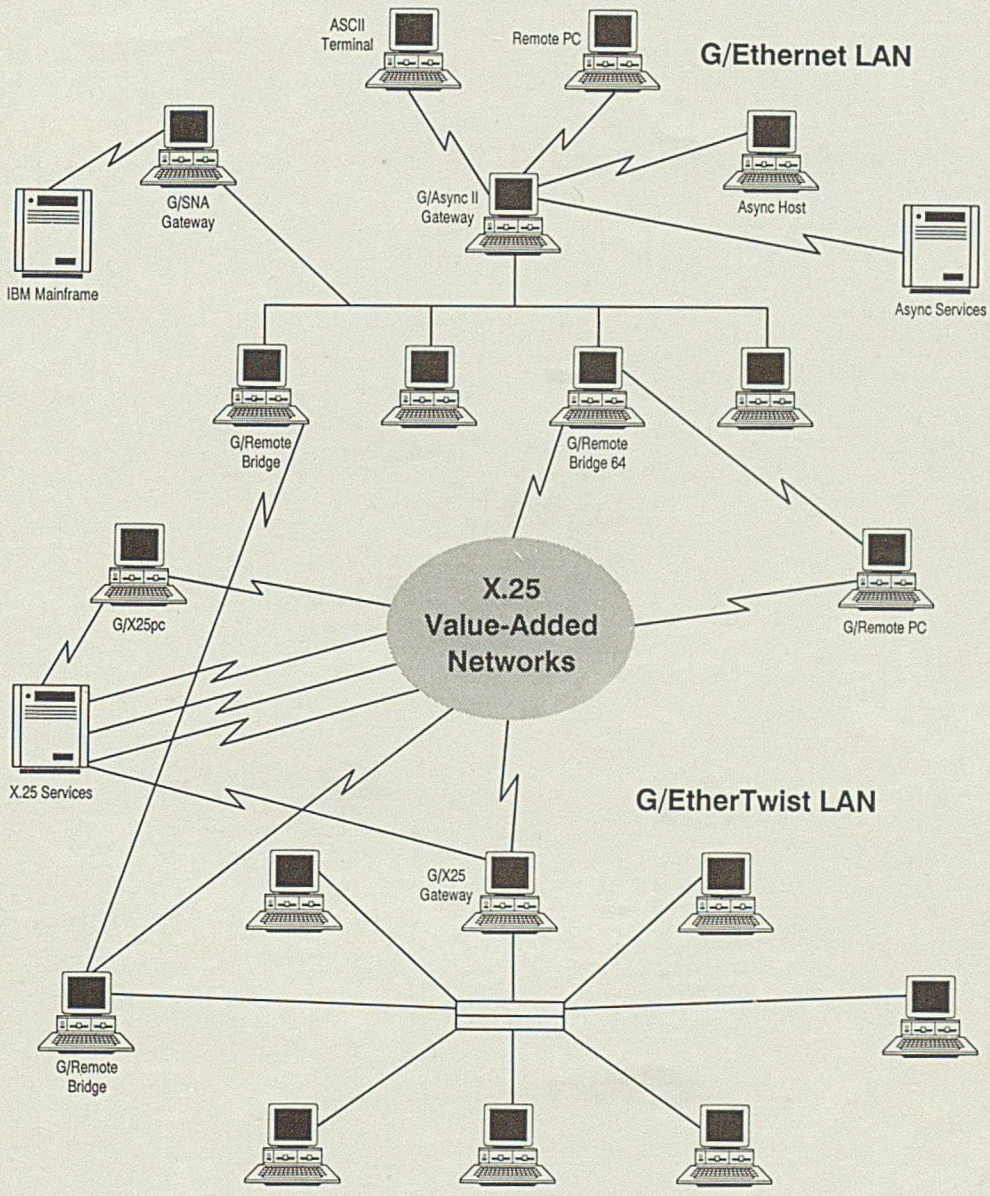
Rozwój Singapuru, którego powierzchnia wynosi zaledwie 600 km² (przy 2,7 mln ludności) polega również na wykorzystaniu możliwości zewnętrznych. Przykładem tego jest tworzenie struktur gospodarczych z przyległymi obszarami Malezji (Jahor) oraz Indonezji (wyspy Riau). Rozwija się tu wiele firm elektronicznych, takich jak Western Digital, Thomson i Sumitomo Electric. Inwestycje zagraniczne Singapuru sięgają już kwoty 1,5 mld dolarów.

JAN RYŻKO

Chrząszcz J.R., Mazur G.B.: Laboratoryjny system mikroprocesorowy INFORMATYKA 1992, nr 6, s. 1 Charakterystyka modułowego laboratoryjnego systemu mikroprocesorowego do celów dydaktycznych i badawczych, zrealizowanego w Instytucie Informatyki i Politechniki Warszawskiej.	**Chrząszcz J.R., Mazur G.B.: Laboratory microprocessor system** INFORMATYKA 1992, No. 6, p. 1 Characteristics of the modular laboratory microprocessor system for didactic and research purposes, which was realized in Informatics Institute of Warsaw University of Technology.	**Chrząszcz J.R., Mazur G.B.: Ein Laboratoriumsmikroprozessorsystem** INFORMATYKA 1992, Nr. 6, S. 1 Eine Charakteristik des Laboratoriumsmikroprozessorsystems für Lehr- und Forschungsaufgaben, das im Institut für Informatik der Warschauer Technischen Universität realisiert wurde.
Wrzeszcz Z.: Inteligentny system gromadzenia informacji faktograficznej INFORMATYKA 1992, nr 6, s. 9 Algorytmy, struktura i przykłady działania systemu gromadzenia informacji faktograficznej GFK, umożliwiającego nie tylko dokumentowanie, ale również badanie przyczyn zdarzeń oraz ich prognozowanie.	**Wrzeszcz Z.: Intelligent system for acquisition of factographic information** INFORMATYKA 1992, No. 6, p. 9 Algorithms, structure and operation examples of the GFK system for acquisition of factographic information, which makes possible not only documenting, but also fact reasons researching and their forecasting.	**Wrzeszcz Z.: Intelligentes System zum Versammeln der faktografischen Information** INFORMATYKA 1992, Nr. 6, S. 9 Algorithmen, Struktur und Beispiele von Betrieb des GKK-Systems zum Versammeln der faktografischen Information, das nicht nur Dokumentierung, sondern auch Ursachenforschung der Ereignisse und ihre Vorhersage ermöglicht.
Caban D., Jarnicki J.: Rekonfiguralny filtr obrazów cyfrowych INFORMATYKA 1992, nr 6, s. 15 Rozwiązanie sprzętowe, zapewniające uzyskanie odfiltrowanego obrazu cyfrowego w czasie rzeczywistym oraz łatwą modyfikację algorytmu filtracji.	**Caban D., Jarnicki J.: Reconfigurable filter for digital images** INFORMATYKA 1992, No. 6, p. 15 Hardware solution which ensures to achieve a real-time filtrated digital image and easy modification of the filtration algorithm.	**Caban D., Jarnicki J.: Rekonfigurierbarer Filter von Digitalbildern** INFORMATYKA 1992, Nr. 6, S. 15 Hardwarelösung, die die Erreichung von abgefiltertem Digitalbild in Echtzeitbetrieb, sowie einfache Modifizierung des Filtrationsalgorithmus ermöglicht.
Piecha J., Przybylski A., Zyguła J.: APW – graficzne narzędzia programowania INFORMATYKA 1992, nr 6, s. 20 Szczegółowa charakterystyka pakietu APW (*Authorware Professional for Windows*), ułatwiającego tworzenie na komputerach klasy IBM PC w środowisku systemu Windows 3.0 programów wspomagających procesy nauczania, a także porównanie APW z innymi pakietami tego typu.	**Piecha J., Przybylski A., Zyguła J.: APW – graphic programming tools** INFORMATYKA 1992, No. 6, p. 20 Detailed characteristics of the APW (Authorware Professional for Windows) package, which helps creation on IBM PC category computers in Window 3.0 environment programs supporting learning process, as well as APW comparison with other similar packages.	**Piecha J., Przybylski A., Zyguła J.: APW – grafische Hilfsmittel der Programmierung** INFORMATYKA 1992, Nr. 6, S. 20 Eine detailierte Charakteristik des APW (Authorware Professional for Windows)-Paketts, das die Herstellung auf Computers von IBM PC-Kategorie in der Windows 3.0-Umwelt von Programmen zur Unterstützung der didaktischen Prozesse erleichtert, sowie eine Vergleichung des APW mit anderen Paketten solcher Art.

Produkty sieciowe LAN i WAN

Gateway communications, inc.



BEZKONKURENCYJNE PRODUKTY
SIECI KOMPUTEROWYCH NAGRADZANE PRZEZ:

PC MAGAZINE, LAN MAGAZINE, INFO WORLD

OFERUJE AUTORYZOWANY DYSTRYBUTOR:

MIKROB

P.W.P.T. MIKROB SP. Z O.O.

20-346 Lublin, ul. Długa 5

Tel. (0-81) 420-61, faks 415-43, teleks 643776 MIKRO PL

PC MAGAZINE EDITORS' CHOICE January 31, 1989 G/Ethernet AT

PC MAGAZINE EDITORS' CHOICE October 16, 1990 G/EtherTwist

0/15/91

Na życzenie wysyłamy katalog produktów. Atrakcyjny program współpracy dla dealer'ów.